2017

Monitoring and Control Framework for Advanced Power Plant Systems Using Artificial Intelligence Techniques

Ghassan Khalil Ismaeel Al-Sinbol

Monitoring and Control Framework for Advanced Power Plant Systems Using Artificial Intelligence Techniques

Ghassan Khalil Ismaeel Al-Sinbol

Dissertation submitted to the
Benjamin M. Statler College of Engineering & Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy
in
Mechanical Engineering

Debangsu Bhattacharyya, Ph.D.
Fernando V. Lima, Ph.D.
Hever Moncayo, Ph.D.
Victor H. Mucino, Ph.D.
Mario G. Perhinschi, Ph.D., Chair

Department of Mechanical and Aerospace Engineering
Morgantown, West Virginia
May 2017

Keywords: Power Plant Monitoring and Control, Artificial Intelligence Techniques, Abnormal Conditions Detection, Identification and Evaluation, Adaptive Control, Optimization

Abstract

Monitoring and Control Framework for Power Plant Using Artificial Intelligence Techniques

Ghassan Al-Sinbol

This dissertation presents the design, development, and simulation testing of a monitoring and control framework for dynamic systems using artificial intelligence techniques. A comprehensive monitoring and control system capable of detecting, identifying, evaluating, and accommodating various subsystem failures and upset conditions is presented. The system is developed by synergistically merging concepts inspired from the biological immune system with evolutionary optimization algorithms and adaptive control techniques.

The proposed methodology provides the tools for addressing the complexity and multi-dimensionality of the modern power plants in a comprehensive and integrated manner that classical approaches cannot achieve. Current approaches typically address abnormal condition (AC) detection of isolated subsystems of low complexity, affected by specific AC involving few features with limited identification capability. They do not attempt AC evaluation and mostly rely on control system robustness for accommodation. Addressing the problem of power plant monitoring and control under AC at this level of completeness has not yet been attempted.

Within the proposed framework, a novel algorithm, namely the partition of the universe, was developed for building the artificial immune system self. As compared to the clustering approach, the proposed approach is less computationally intensive and facilitates the use of full-dimensional self for system AC detection, identification, and evaluation. The approach is implemented in conjunction with a modified and improved dendritic cell algorithm. It allows for identifying the failed subsystems without previous training and is extended to address the AC evaluation using a novel approach.

The adaptive control laws are designed to augment the performance and robustness of baseline control laws under normal and abnormal operating conditions. Artificial neural network-based and artificial immune system-based approaches are developed and investigated for an advanced power plant through numerical simulation.

This dissertation also presents the development of an interactive computational environment for the optimization of power plant control system using evolutionary techniques with immunity-inspired enhancements. Several algorithms mimicking mechanisms of the immune system of superior organisms, such as cloning, affinity-based selection, seeding, and vaccination are used. These algorithms are expected to enhance the computational effectiveness, improve convergence, and be more efficient in handling multiple local extrema, through an adequate balance between exploration and exploitation.

The monitoring and control framework formulated in this dissertation applies to a wide range of technical problems. The proposed methodology is demonstrated with promising results using a high validity Dynsim® model of the acid gas removal unit that is part of the integrated gasification combined cycle power plant available at West Virginia

University AVESTAR Center. The obtained results show that the proposed system is an efficient and valuable technique to be applied to a real world application. The implementation of this methodology can potentially have significant impacts on the operational safety of many complex systems.

Dedication

To the loving memory of my father, gone but never forgotten,

To my mother, you are the best mother someone could ask for,

To my wife, Haneen, for being with me through the difficult times,

To my brothers and sisters, for their enormous love and support,

To my kids, Yousif and Layan, for the fun and the crazy times.

Acknowledgements

It is a great pleasure to take this opportunity to thank my advisor, Dr. Mario G. Perhinschi for his support, guidance, patience, and teachings throughout my education. Without his true mentoring, support, and patience, this work would not have been possible.

I would also like to express my sincere thanks to the members of my committee, Dr. Victor Mucino, Dr. Debangsu Bhattacharyya, Dr. Fernando V. Lima, and Dr. Hever Moncayo, for their encouragement and insightful comments I received during the preparation of this dissertation. Thanks are also due to Dr. Dia Al Azzawi for the outstanding help I received from him.

I thankfully acknowledge the financial support provided by the Department of Energy (DOE) and the National Energy Technology Laboratory (NETL) who sponsored this project through grant no. DE-FE0012451 titled “Development of Integrated Biomimetic Framework with Intelligent Monitoring, Cognition and Decision Capabilities for Control of Advanced Energy Plants”.

Finally, I am truly thankful to my wife, Haneen Hanoon, for her love and support during my study. I also would like to take the opportunity to thank Dr. Mohammed Khalil and his wife Mrs. Raja Khalil for their unlimited encouragement. My particular thanks go to my father in law Mr. Kareem Hanoon and my mother in law Ms. Seddeeqah Rajab for their great help, support, and advice. I also would like to thank Bilal Salih, Ahmed Hanoon, Jessica Siqueira, and all my friends who have taken care of me and helped me through the years of my doctoral study.

Table of Contents

Abstract ..... **ii**

Dedication ..... **iv**

Acknowledgements ..... **v**

List of Figures ..... **ix**

List of Tables ..... **xiii**

Nomenclature ..... **xiv**

Chapter 1. Introduction **1**

1.1. Background 1
1.2. Research Objectives 3
1.3. Contributions 3
1.4. Dissertation Outline 5

Chapter 2. Literature Review **6**

2.1. An Overview of the Biological Immune System 6
2.2. An Overview of the Artificial Immune System 10
2.2.1. AIS for Abnormal Conditions Management 10
2.2.2. AIS for Control 13
2.2.3. AIS for Optimization 15

Chapter 3. AIS-Based Framework for Monitoring and Control **17**

3.1. General Architecture of the AIS-Based Monitoring and Control Framework 17
3.2. AIS – Definitions and Concepts 20
3.3. ACDIE Problem Formulation 20
3.4. Accommodation Problem Formulation 23
3.5. Evolutionary Optimization Problem Formulation 24

Chapter 4. The Partition of the Universe Approach **26**

4.1. Self / Non-self Discrimination 26
4.2. Preliminary Data Preparation 26
4.3. The Self Generation Using Data Clustering Approach 27
4.4. The Self Generation Using Partition of Universe Approach 29
4.5. PUA vs. DCA 33
4.5.1. Algorithm Parameters 33
4.5.2. Computational Issues 34
4.5.3. Comparison Summary 36

Chapter 5. AIS-based Abnormal Condition Detection, Identification, and Evaluation 37

5.1. Self / Non-self Discrimination 37
5.2. The Artificial Dendritic Cell Mechanism 39
5.2.1. The Artificial DC Inputs 40
5.2.2. The Artificial DC Components 41
5.2.3. The Artificial DC Algorithm 46
5.3. Abnormal Conditions Detection 46
5.4. The Naïve Bayes Classifier 49
5.5. Abnormal Conditions Identification 50
5.5.1. The Subsystem Pattern Approach 50
5.5.2. The Features Pattern Approach 51
5.6. Abnormal Conditions Evaluation 53
5.6.1. The Partition Tracking Pattern Approach 54
5.6.2. The Feature Pattern Approach 56
5.7. One Step AC Identification and Evaluation 57
5.8. ACDIE Performance Indices 58

Chapter 6. Abnormal Conditions Accommodation 62

6.1. The Artificial Neural Net Adaptive Controller 63
6.2. AIS-Based Adaptive Controller 67

Chapter 7. AIS Based Evolutionary Optimization 73

7.1. The Genetic Algorithm 73
7.2. The Genetic Algorithm Description 74
7.2.1. Initial Population 74
7.2.2. Fitness Function 77
7.2.3. Parent Population Selection 77
7.2.4. Mutation 79
7.2.5. Crossover 80
7.3. Immunity-Enhanced Genetic Algorithm 81
7.3.1. Seeding and Vaccination 82
7.3.2. Clonal Selection Loop 83
7.3.3. New Individuals Addition 85
7.3.4. Affinity-based Selection Algorithm 86

Chapter 8. The Acid Gas Removal Unit 88

8.1. The Acid Gas Removal Unit Description 88
8.1.1. The H_2S Absorber 89
8.1.2. The H_2S Stripper 90
8.1.3. The CO_2 Absorber 90

8.1.4. The SELEXOL Makeup Flow 91
8.2. The Acid Gas Removal Unit Model 91
8.3. The Acid Gas Removal Unit Nominal Operations 92
8.4. The Acid Gas Removal Unit Abnormal Conditions 92

Chapter 9. Results and Discussions 94

9.1. Demonstration of ACDIE Scheme 94
9.1.1. AC Detection Performance 97
9.1.2. AC Identification Performance 99
9.1.3. AC Evaluation Performance 100
9.2. Demonstration of Adaptive Control Mechanisms 105
9.2.1. Demonstration Using Linearized Model 105
9.2.2. Demonstration Using Non-Linear Model 111
9.2.3. Demonstration Using the Hyper System 113
9.3. Demonstration of Optimization Scheme 120

Chapter 10. Developed Tools 124

10.1. The Self/Non-self Visualization Tool 124
10.2. The ACDIE Interface 126
10.3. The Evolutionary Optimization Environment Interface 128

Chapter 11. Conclusion and Future Work 132

Chapter 12. References 134

List of Figures

Figure 2-1. Antigen Presenting Cells 7
Figure 2-2. Positive Selection Process 8
Figure 2-3. Negative Selection Process 9
Figure 3-1. Power System Biomimetic Monitoring and Control Process 18
Figure 3-2. General Framework Architecture 19
Figure 3-3. Targeted System and Its Components 21
Figure 3-4. AC Active Accommodation 23
Figure 4-1. Data Preparation for Self Generation 27
Figure 4-2. K-means Algorithm for Data Clustering 28
Figure 4-3. Sample 2-D Self Projection Generated Using K-means [91] 28
Figure 4-4. The Partition of Universe Approach Algorithm 31
Figure 4-5. Self Representation for PUA 32
Figure 4-6. Sample 2-D Self Generated Using PUA (Uniform Square Grid) 32
Figure 4-7. Sample 2-D Self Generated Using PUA (Uniform Hexagon Grid) 33
Figure 5-1. The Artificial DC Data Structure 42
Figure 5-2. The Artificial DC Algorithm 47
Figure 5-3. The Artificial DC Algorithm for AC Detection 48
Figure 5-4. Training the Naïve Bayes for AC Identification 53
Figure 5-5. Online AC Identification Using Features Pattern Approach 53
Figure 5-6. Training the Naïve Bayes for AC Evaluation 55
Figure 5-7. Online AC Evaluation Using Partition Tracking Pattern Approach 56
Figure 5-8. Training the Naïve Bayes for One Step AC Identification and Evaluation 57
Figure 5-9. Online One Step AC Identification and Evaluation 58
Figure 6-1. Architecture of the Biomimetic Adaptive Control Laws 62
Figure 6-2. Adaptive Control Based on an Artificial Neural Network Mechanism 63
Figure 6-3. Single Hidden Layer ANN 66
Figure 6-4. Adaptive Control Based on an AIS Mechanism 67

Figure 6-5. Humoral Immune System Feedback Mechanism 69
Figure 6-6. AIS-based Adaptive Mechanism 72
Figure 7-1. Overview of the GA 75
Figure 7-2. Chromosome Structure for Gain Optimization 75
Figure 7-3. Chromosome Structure for Setpoint Optimization 76
Figure 7-4. Roulette-Wheel Selection Approach 78
Figure 7-5. The Mutation Operation 80
Figure 7-6. The Single Point Cross Operation 81
Figure 7-7. The Single Point Cross Algorithm 81
Figure 7-8. Immunity-Enhanced Genetic Algorithm Implementation 82
Figure 7-9. Seeding and Vaccination of Initial Population 83
Figure 7-10. Clonal Selection Loop 84
Figure 7-11. New Individual Generation Based on Low Affinity to Self 86
Figure 7-12. Affinity-based Individual Selection 87
Figure 8-1. Schematic of the AGR Unit [112] 89
Figure 8-2. Dynsim®-Matlab® Engine Link Diagram 92
Figure 9-1. Reference Partition Tracking Pattern for AC1 and AC2 100
Figure 9-2. Reference Partition Tracking Pattern for AC3 and AC4 101
Figure 9-3. Reference Partition Tracking Pattern for AC5 and AC6 101
Figure 9-4. Reference Partition Tracking Pattern for AC7 101
Figure 9-5. Reference Partition Tracking Pattern for AC8 102
Figure 9-6. Reference Partition Tracking Pattern for AC9 102
Figure 9-7. Reference Partition Tracking Pattern for AC10 102
Figure 9-8. Reference Partition Tracking Pattern for AC11 103
Figure 9-9. Reference Partition Tracking Pattern for AC12 103
Figure 9-10. Reference Partition Tracking Pattern for AC13 and AC14 103
Figure 9-11. Adaptive Control Testing Using PID Controller and a Linearized Model 106
Figure 9-12. Optimal Trajectory Test with No Adaptive Augmentation 108

Figure 9-13. Optimal Trajectory Test with ANN-Plant-State-based Augmentation 108

Figure 9-14. Optimal Trajectory Test with ANN-Plant Output-based Augmentation 109

Figure 9-15. Optimal Trajectory Test with Immunity-based Augmentation 109

Figure 9-16. Standalone BIOCS Configuration 111

Figure 9-17. BIOCS and Adaptive Control Configuration 111

Figure 9-18. CO_2 Percentage in Outgoing Stream Tracking Using BIOCS 112

Figure 9-19. Temperature of Recycled Solvent Tracking Using BIOCS 112

Figure 9-20. CO_2 Percentage in Outgoing Stream Tracking Using BIOCS and ANN 113

Figure 9-21. Temperature of Recycled Solvent Tracking Using BIOCS and ANN 113

Figure 9-22. Fuel Cells/Gas Turbine Hyper System [119] 114

Figure 9-23. Hyper System Baseline Control Laws 115

Figure 9-24. Response to TS Step Input Under Nominal Conditions 118

Figure 9-25. Response to CA Step Input Under Nominal Conditions 118

Figure 9-26. TS Response to FV Disturbance Under Nominal Conditions 119

Figure 9-27. CA Response to HAB Disturbance Under Nominal Conditions 119

Figure 9-28. Response to TS Step Input Under Abnormal Conditions 119

Figure 9-29. Standard Genetic Algorithm - Variation of Performance Index of the Best Individual and the Population Average 121

Figure 9-30. Evolutionary Optimization Algorithm with Clonal Loop - Performance Index of the Best Individual and the Population Average 121

Figure 9-31. Evolutionary Optimization Algorithm with Vaccination - Performance Index of the Best Individual and the Population Average 122

Figure 9-32. Evolutionary Optimization Algorithm with Seeding - Performance Index of the Best Individual and the Population Average 123

Figure 9-33. Variation of the Solvent Temperature Using the Best Solution 123

Figure 10-1. AIS Selves and Non – Selves 2-D Projections Viewer 124

Figure 10-2. GUI for Parameter Setting of the DC Mechanism 126

Figure 10-3. Load Full Command Request Message Box 127

Figure 10-4. Load Initial Conditions Request Message Box 127

Figure 10-5. ACDIE Interactive GUI 127

Figure 10-6. GUI for Control System Optimization Problem and Algorithm Selection 129

Figure 10-7. GUI for GA Initial Population and Convergence Parameter Setting 130

Figure 10-8. GUI for Setup of Parameter of Clonal Selection Loop 130

Figure 10-9. GUI for Setup of GA Selection Method and Alteration Parameters 131

Figure 10-10. GUI for Output Display 131

List of Tables

Table 4-1. Sample Computational Time for PUA and DCA....35

Table 8-1. The AGR Unit Subsystems....88

Table 8-2. AGR Unit Normal Operational Constraints [116]....92

Table 8-3. AGR Unit ACs List....93

Table 9-1. The AGR Selected Features....95

Table 9-2. The Targeted Subsystems and Resulted Selves Dimensions....97

Table 9-3. AC Detection Performance....98

Table 9-4. AC Identification Performance....99

Table -9-5. AC Evaluation Performance....104

Table 9-6. Tracking Performance of PID and Different Adaptive Controllers....107

Table 9-7. Tracking Performance of Biomimetic and Different Adaptive Controllers....110

Table 9-8. Hybrid System Linear Transfer Functions Parameters....115

Table 9-9. Parameter Range for Abnormal Condition Simulation....116

Table 9-10. Composite Performance Index (PI) Under Nominal Conditions....117

Nomenclature

Alphabetical

C	Self cluster
CSM_t	Co-stimulatory molecules
D_t	Discrimination matrix
$\overline{D}_t$	Complementary discrimination matrix
Det	Detection outcome
DR	Detection rate
$EvDQ$	Direct quantitative evaluation outcome
$EvIQ$	Indirect quantitative evaluation outcome
EvQ	Qualitative evaluation outcome
F_0	Non-triggered feature matrix
F_1	Triggered features matrix
$\mathcal{F}$	Set of features
$\overline{\mathcal{F}}$	Normalized set of features
FA	False alarm
FF	Fitness function
$\mathcal{FP}$	Feature point location on universe grid
i	Feature index
Idt	Identification outcome
$IL10$	Interleukin-10
$IL12$	Interleukin-12
IR	Identification Rate
K	Number of stimulatory T-cells
$\widetilde{K}$	Number of residual cytotoxic T-cells
k	Subsystem index
K_D	Derivative gain
K_I	Integral gain
K_P	Proportional gain
$\mathcal{L}$	DC cell life
l	Self projection index
$\mathcal{M}$	DC migration threshold
N	Number of features
N_{MDC}	Number of migrated DCs
N_p	Number of lower-dimensional self projections
N_s	Number of subsystems
P	Raw self-data point
$\overline{P}$	Normalized raw self-data point
PE	Performance vector
$\overline{PE}$	Normalized performance vector
PT	Partition tracking matrix
PTP	Partition tracking pattern
PW	Performance vector weight
Q	Self/non-self discrimination outcome

$\overline{Q}$	Complementary self/non-self discrimination outcome
R	Number of suppressor T-cells
S	Self
$\hat{S}$	Non-self
SER	Severity evaluation rate
SF	Self projections-features mapping matrix
t	Current sample time
TER	Type evaluation rate
UP	Number of uniform projections
W_0	Non-triggered matrix self confidence vector
W_1	Triggered matrix self confidence vector
W_s	Sample time confidence vector

Greek

α_{il}	Self projections-features mapping matrix elements
Δp	Partition location change
Γ_{10}	Interleukin-10 accumulation functions
Γ_{12}	Interleukin-12 accumulation functions
λ_s	DC selection flag
Π	Self partitions resolution vector
π_i	Feature i^{th} self partition resolution
πe_i	Feature i^{th} evaluation partition resolution
ψ_{il}^0	Non-triggered features matrix elements
ψ_{il}^1	Triggered features matrix elements
Σ	Sample covariance matrix
σ	Sample standard deviation
τ	Detection window time vector
φ_i	Feature i value
$\overline{\varphi}_i$	Feature i normalized value

Acronyms

AC	Abnormal Conditions
ACDIE	Abnormal Condition Detection, Identification, and Evaluation
ACDIEA	Abnormal Condition Detection, Identification, Evaluation, and Accommodation
ACM	Abnormal Conditions Management
AGR	Acid Gas Removal
AIS	Artificial Immune System
ANN	Artificial Neural Net
APC	Antigen Presenting Cell
AVESTAR	Advanced Virtual Energy Simulation Training and Research Center
BIOCS	Biologically-Inspired Optimal Control Strategy
BMC	Biomimetic Monitoring and Control
DC	Dendritic Cell
DCA	Data Clustering Approach
DOE	Department Of Energy

EA	Evolutionary Algorithm
EOE	Evolutionary Optimization Environment
GA	Genetic Algorithm
GUI	Graphical User Interface
HMS	Hierarchical Multi-Self
IC	Initial Conditions
IGCC	Integrated Gasification Combined Cycle
LF	Load Full
MHC	Major Histocompatibility Complex
NETL	National Energy Technology Laboratory
PI	Performance Index
PUA	Partition of the Universe Approach
WVU	West Virginia University

Chapter 1. Introduction

1.1. Background

Increasingly strict environmental regulations, safety concerns, and economic objectives significantly expand modern power plants complexity and multi-dimensionality [1]. These plants are expected to function at maximum efficiency under both normal and abnormal operational conditions. Handling such a challenging task requires advanced intelligent monitoring and decision-making capabilities as part of a plant-wide control strategy [2]-[4]. Faults occurrence in power plants can cause losses in efficiency, equipment damage, and unsafe operation conditions. It is important to diagnose these faults so that necessary actions for mitigation can be taken and/or maintenance can be accordingly planned in advance. Power plant subsystem faults could result from various sources such as stuck control valve, malfunctioning sensors, solid deposits, structural damage, etc.

Detection of process specific abnormal conditions has been addressed in the literature [5]-[7], typically for low complexity systems, and low-order abnormal conditions that only require few features for their detection. Most of the fault detection algorithms require a process model. A process model consists of mathematical equations that represent the system's physical phenomena. It is worth mentioning that formulation of process models can be very difficult and time consuming. Various abnormal condition detection approaches have been proposed, such as artificial neural networks [8], [9], support vector machines classifiers [10], [11], sensitive principal component analysis [12], Kalman filters [13], self-organizing maps [14], and others [15]. The selection of minimal sets of sensors as a pre-requisite for fault detection has also been investigated [16], [17]. Some of the techniques can detect faults very quickly, but cause high number of false alarms because of their high sensitivity to changes in the system. Some of the approaches are robust to noise and uncertainties in the system, but have lower detection rate for certain faults. For a comprehensive and integrated solution covering the extreme diversity of possible abnormal conditions, more powerful tools are needed.

The terms "failure" and "fault" are broadly used in the literature to indicate that a system is working outside the intended operating conditions. In this dissertation, the generic term of ***abnormal conditions (AC)*** will be used to refer to any departure from

a reference or normal condition due to any subsystem faults or any other situation that requires particular attention for safety purposes. The process of acknowledging the presence of the AC, isolating the primarily affected subsystem(s) or the source of the AC, and assessing the nature and severity of the AC will be referred to as ***abnormal conditions detection, identification, and evaluation (ACDIE).***

Finding a comprehensive solution to the ACDIE problem is an exceptionally complex, multidimensional task that requires appropriate tools, high-level accuracy, and extensive robustness, while performed in a timely manner. The ACDIE performance assessment must consider minimum false alarms under normal operating conditions and reduced detection time, high detection rate, identification rate, and evaluation rate under AC. Unreasonable delays in detecting failures, usually lead to undesired consequences such as stall events, loss of control, and severe vibrations. The ideal ACDIE process must also be capable of detecting unknown failures and not misclassifying them as one of the known faults or as normal operation. It should also be adaptive to system changes, robust to system disturbances and uncertainties, and scalable to dimensionality changes of the system. In addition, an ideal ACDIE should be able to detect and correctly identify multiple failures when they coexist in the system.

Engineers sought outside of the standard framework of control systems for ideas to address such complicated problem. Biologically-inspired methodologies have recently become very popular among researchers, and various approaches have been successfully developed and implemented. Immunity-based techniques are extremely promising candidates for solving the ACDIE problem. The biological immune system exhibits every requirement that ACDIE problem is restrained to: the capability of detecting any harmful intruders, identifying the attacker, assessing the level of danger, accommodating by generating antibodies to fight the intruders, and memorizing the attacker for much faster and more efficient defense in future encounters.

The ***artificial immune system (AIS)*** [18] has emerged in recent years as a new artificial intelligence computational paradigm. The concept has shown a very promising potential for a variety of applications including ACDIE. Immunity inspired methodologies have shown to provide the intelligent tools capable of gathering the information about the existence of a failed subsystem, the nature of the current fault, and the severity of the fault

as soon as it takes place such that an accommodation strategy by the control laws could be triggered. In fact, this information is highly relevant to the operators as well, increasing their situational awareness.

1.2. Research Objectives

The objective of this research aims at designing an intelligent monitoring system powered with cognition and decision capabilities that mimic the artificial immune system. The ***abnormal conditions management (ACM)*** process presented in this dissertation is composed of four steps: detection, identification, evaluation, and accommodation. The AC detection is the process of detecting the failure in the targeted system (i.e. power plant) as soon as it takes place. The AC identification is the process of isolating the failed subsystem. The AC evaluation can be of a qualitative or quantitative nature. The qualitative evaluation is the assessment of the failure type. The quantitative evaluation assesses the AC severity and its impact on the system. The AC accommodation and controller optimization is the process of adapting the system controllers to the current AC. The methodologies and algorithms developed in this research are tested in the plant-wide model of the acid gas removal unit as part of the ***integrated gasification combined cycle (IGCC)*** that is available in the ***Advanced Virtual Energy Simulation Training and Research Center (AVESTAR)*** [19] at ***West Virginia University (WVU)***.

1.3. Contributions

The main contribution of this research effort is the extensive use, for the first time, of immunity-inspired techniques to address advanced power plant operations. This was achieved by bringing the following major contributions:

- formulation of immunity inspired techniques to address advanced power plant health monitoring and control problem
- design and implementation of the partition of the universe approach as a novel data clustering approach for self/non-self generation
- modifying an artificial DC algorithm for AC detection, identification, and evaluation and applying it to power plants
- extending the partition of the universe approach concept to address the AC evaluation problem
- the development of a novel biomimetic adaptive approaches to augment baseline power plant control laws for increased robustness under ACs

- developing immunity based optimization enhancements for genetic algorithms

The work related to the research effort presented in this study has resulted in the following publications:

Journals papers

1. Perhinschi, M., **Al-Sinbol, G.,** Bhattacharyya, D., Lima, F., Mirlekar, G., Turton, R. Development of an Immunity-based Framework for Power Plant Monitoring and Control. *Advanced Chemical Engineering Research,* 4(1), 2015
2. **Al-Sinbol, G**., Perhinschi, M. Generation of power plant artificial immune system using the partition of the universe approach. *International Review of Automatic Control (IREACO)* 9(1), 2016
3. Perhinschi, M., **Al-Sinbol, G.** Artificial dendritic cell algorithm for advanced power system monitoring. *International Review of Automatic Control (IREACO)* 9(5), 2016
4. **Al-Sinbol, G.,** Perhinschi, M., Bhattacharyya, D., Evolutionary Optimization of Power Plant Control System Using Immunity-inspired Algorithms. *International Review of Chemical Engineering (I.RE.CH.E.),* 9(1), 2017
5. **Al-Sinbol, G.,** Perhinschi, M., Development of an Artificial Immune System for Power Plant Abnormal Condition Detection, Identification, and Evaluation, submitted to *International Review of Automatic Control (I.RE.A.CO.),* Feb 2017
6. **Al-Sinbol, G.,** Perhinschi, M., Pezzini, P., Bryden, K., Tucker, D., Investigation of Biomimetic Adaptive Mechanisms for Hybrid Power Plant Control, to be submitted to *International Review of Automatic Control (I.RE.A.CO.),* May 2017

Conference papers, presentations, and posters

1. Bhattacharyya D., Turton R., Lima F., Perhinschi M.G., Bankole T., Mirlekar G., **Al-Sinbol G**., Gebreslassie B. H.,Diwekar U., Development of Integrated Biomimetic Framework with Intelligent Monitoring, Cognition, and Decision Capabilities for Control of Advanced Energy Plants, *Presented at 2015 NETL Crosscutting Research Review Meeting, Pittsburgh, PA, April 27-30, 2015*
2. Bhattacharyya D., Turton R., Lima F., Perhinschi M.G., Bankole T., Mirlekar G., **Al-Sinbol G**., Gebreslassie B. H.,Diwekar U., Development of Integrated Biomimetic Framework with Intelligent Monitoring, Cognition, and Decision Capabilities for Control of Advanced Energy Plants, *Presented at 2016 NETL Crosscutting Research & Rare Earth Elements Portfolios Review Meeting, Pittsburgh, PA, April 18-22, 2016*
3. **Al-Sinbol, G.,** Perhinschi, M., Bhattacharyya, D., Evolutionary Optimization Environment for Power Plant Control with Dynsim® Interface. *Presented at 2016 AIChE Annual Meeting, San Francisco, CA, November 13-18, 2016*
4. **Al-Sinbol, G.,** Perhinschi, M., Bhattacharyya, D., Power Plant Abnormal Condition Detection Using the Artificial Immune System Paradigm. *Poster Presented at 2016 AIChE Annual Meeting, San Francisco, CA, November 13-18, 2016*
5. Bhattacharyya D., Turton R., Lima F., Perhinschi M.G., Bankole T., Mirlekar G., **Al-Sinbol G**., Gebreslassie B. H.,Diwekar U., Development of Integrated Biomimetic Framework with Intelligent Monitoring, Cognition, and Decision Capabilities for Control of Advanced Energy Plants, *Presented at 2017 NETL Project Review Meeting for Crosscutting Research, Gasification Systems, and Rare Earth Elements Research Portfolios, Pittsburgh, PA, March 20-23, 2017*

1.4. Dissertation Outline

This dissertation is organized as follows. An overview of the biological and artificial immunity is introduced in Chapter 2 with a survey of research efforts in the area of AIS. Chapter 3 describes the general immunity-based framework for health monitoring, control, and optimization. Chapter 4 introduces a novel approach for generating the technical system self using the partition of the universe. The immunity-based abnormal conditions detection, identification, and evaluation are presented in Chapter 5. Chapter 6 introduces different abnormal condition accommodation approaches. Evolutionary and immunity based optimization algorithms are introduced in Chapter 7. The acid gas removal unit is presented in Chapter 8. The results and discussion are presented in Chapter 9. Chapter 10 presents interactive software tools that were developed as part this research. Future research work and recommendations are offered in Chapter 11.

Chapter 2. Literature Review

Artificial intelligence algorithms were developed to obtain solutions to a broad class of complex problems, which could not be solved by traditional methods. The AIS is a relatively new paradigm in the field of computational artificial intelligence inspired by the biological immune system. This chapter provides an overview of the biological immune system, AIS, and some areas of relevant application of the AIS.

2.1. An Overview of the Biological Immune System

The biological immune system consists of organs, cells, and molecules responsible for protecting the organism from the potentially harmful antigens, such as viruses, bacteria, and other intruders. It has the ability to detect foreign substances, respond adequately to the danger, and keep memory of previous invasions.

The biological immune system defends the living organism from the antigens using different layers of defense. The anatomic barrier makes the body first line of defense against intruders. It prevents the potential invaders from entering using both physical barriers and chemical substances. Skin, saliva, mucous, and tears are considered part of the anatomic barriers [20]. The innate immune system, which is inherited from the ancestors, is the second line of defense against invading pathogens. The innate immune system cells are always active and quickly react in a non-specific way to any class of recognized pathogens. Jawed vertebrates developed a third layer of protection called the adaptive immune system. The adaptive immune is usually inactive, and it is activated by the innate immune system cells. Its response is built through previous exposures to infections. It reacts specifically to pathogens and antigens and possesses immunological memory that allows a quicker response each time a pathogen is subsequently encountered [21].

The biological immune system organs can be classified into central lymphoid organs and peripheral lymphoid organs. The role of central lymphoid organs, which include the bone marrow and the thymus, is to generate and aid mature immune cells. On the other hand, the peripheral lymphoid organs further the interaction between lymphocytes, the white blood cells, and antigens. Peripheral lymphoid organs include lymph nodes, the spleen, and mucosal and submucosal tissues [22].

The biological immune system is made up of a mixture of specialized cells, which interact among themselves to accomplish appropriate immunological responses. Phagocytes are specialized innate immune system cells that capture and process antigens. ***Macrophages (MΦs)*** and ***dendritic cells (DCs)*** are the dominant phagocytes in the immune system. On the other hand, lymphocytes are specialized adaptive immune system cells that are produced in the bone marrow and circulate through the blood and lymph system. T-cells and B-cells form the majority population of lymphocytes [21].

Phagocytes are presented in the peripheral tissues that are in contact with the external environment, such as skin, the inner covering of nose, lungs, stomach, and intestines [20]. Phagocytes have receptors on their surfaces that can detect harmful pathogens. As soon as a pathogen is detected, the phagocyte expands itself around the pathogen and engulfs it. After engulfment, DCs and MΦs are capable of breaking down the pathogen into its constituent molecules or peptides. These peptides are then attached to the cell's ***major histocompatibility complex class II (MHC-II)*** special complex protein, which moves the peptides back to the phagocyte's surface where they can be presented to T-cells [23]. Therefore, DCs, MΦs, and other similar immunity cells are called antigen presenting cells (APCs). Figure 2-1 presents a DC and its interaction with T-cells.

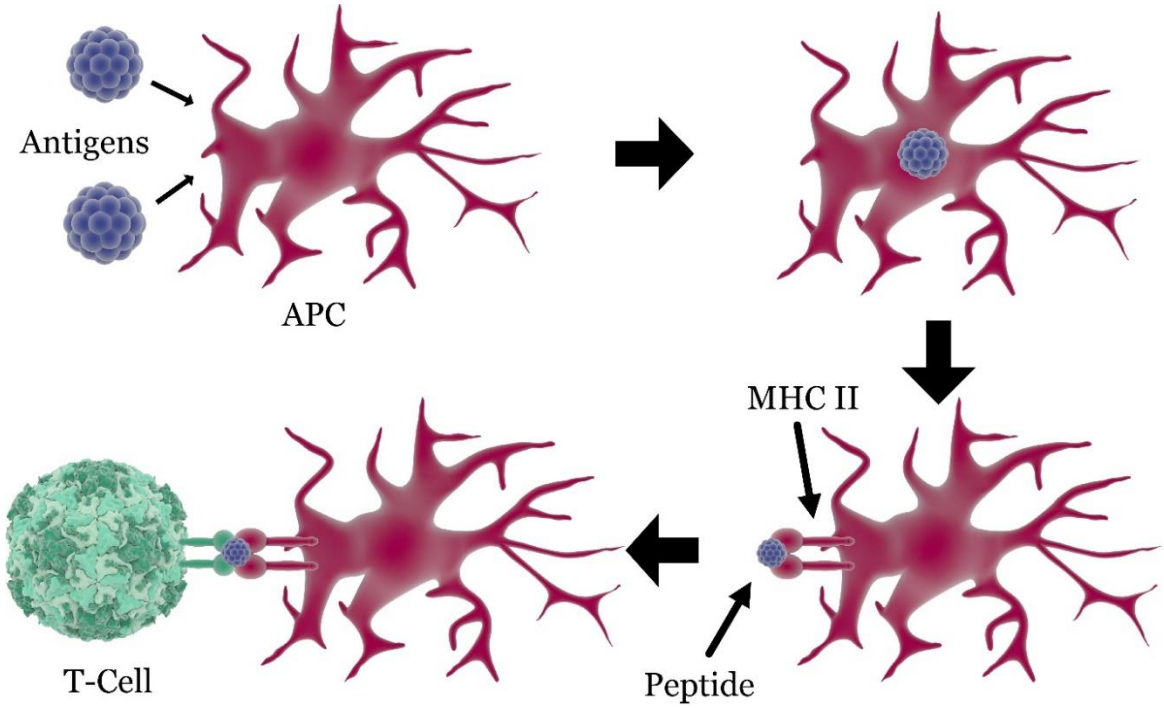

Figure 2-1. Antigen Presenting Cells

The DCs developed from the stem cells in the bone marrow are initially immature cells. Immature DCs move through the blood stream to the peripheral tissues where they can interact with antigens. Immature DCs are good at engulfing antigens; however, they are poor APCs. Once they become mature, DCs become more efficient APCs and begin to migrate to the lymph nodes where they activate T-cells by the antigen presenting process [24].

T-cells are created by the bone marrow and mature in the thymus. In the maturation process, T-cells undergo a two-step maturation process referred to as the positive selection and the negative selection. In the positive selection process, only T-cells that recognize the self-MHC molecules are kept, while those T-cells who fail to recognize the self MHC molecules are eliminated. In the negative selection process, on the other hand, T-cells that bind to the MHC and self-peptide are removed because those T cells might cause a detrimental autoimmune response and only T-cells that do not bind to MHC and self-peptide are kept [22], [25]. A schematic drawing of the positive and negative selection process is presented in Figure 2-2 and Figure 2-3, respectively. In the drawing, the T-Cell body is represented with the bigger circular shape, the MHC pattern is represented by the two outside shapes, and the peptide pattern by the hollow shape in the middle. In the positive selection process a match or a mismatch is considered only for the MHC pattern, as shown using circles in Figure 2-2. The T-cells that mismatch all the MHC patterns are rejected, while those that match undergo the negative selection process. In the negative selection process, a match or mismatch is considered based on both the MHC and the self peptide patterns as shown using circles in Figure 2-3. This time, only T-cells mismatching all available patterns are selected to survive and mature.

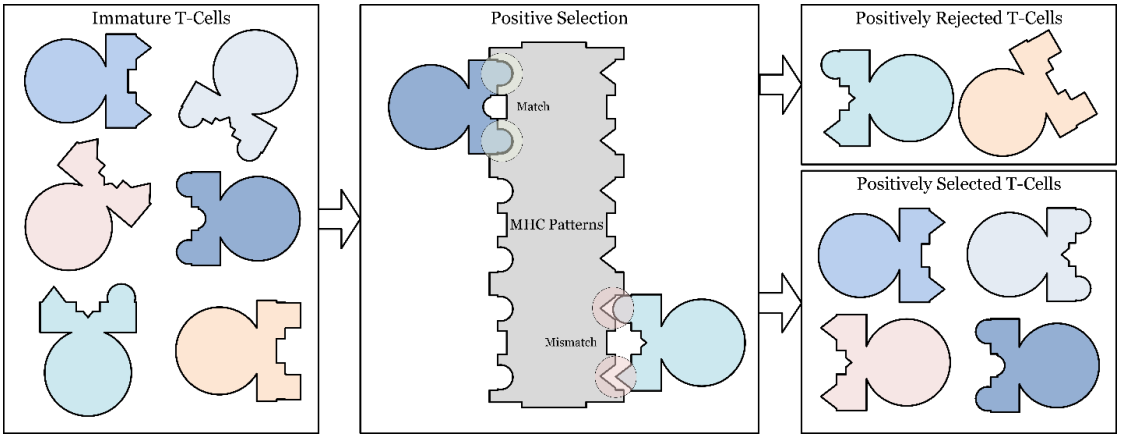

Figure 2-2. Positive Selection Process

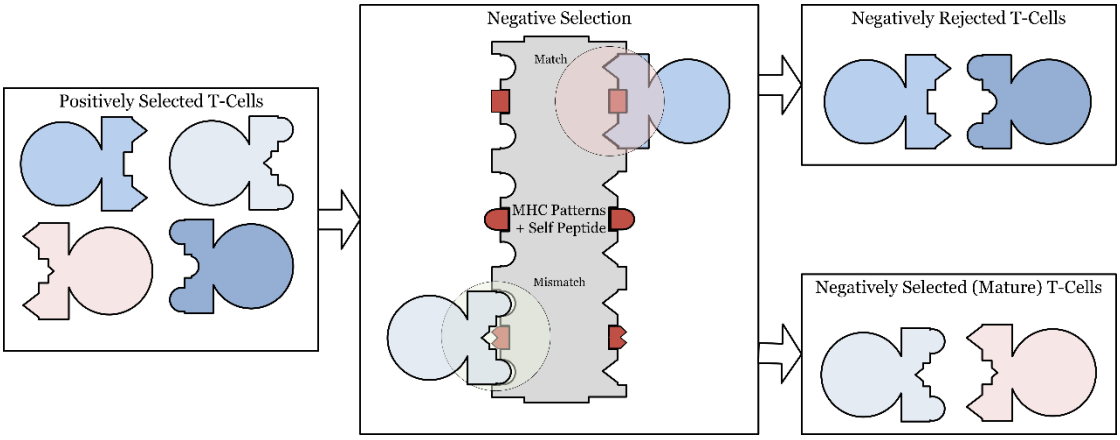

Figure 2-3. Negative Selection Process

There exist different types of T-cells, each having a specific role. The important types are Helper T-cells, Cytotoxic T-cells, Memory T-cells, and Suppressor T-cells. Helper T-cells become activated when they are presented with their specific MHC II-peptide complex by an APC. Once activated, helper T-cells continuously differentiate into memory T-cells and effector T-cells. Memory T-cells stay in the body for decades to help provide a faster response if the same antigen is ever encountered again. Effector T-cells release chemical alarm signals called interleukin-12 that cause particular types of cytotoxic T-cells and B-cells to proliferate [21]. Cytotoxic T-cells, on the other hand, are responsible for eliminating the infected body cells. Once infected by an antigen, cells can also digest and present the antigen peptides on the cell surface using MHC I complex protein. The cytotoxic T-cells that able are to recognize the MHC I-peptide combination displayed by an infected cell, bind to the infected cell and produce chemicals that kill the infected cell and therefore the intruder [20]. The adaptive immune response using the cytotoxic T-cells is called cell-mediated response. The role of the suppressor T-cells is to shut down helper T-cell-mediated immunity toward the end of an immune reaction and to suppress the generation of the cytotoxic T-cells and the antibodies [22]. This is meant to prevent over-reaction and save resources and can be viewed as a regulatory feedback mechanism.

As mentioned earlier, B-cells are another type of primarily specialized lymphocytes immune cells, which were produced and matured in the bone marrow. They are responsible for producing and secreting Y-shaped antibodies, which bind to antigens and mark them for destruction by phagocytes. It is worth mentioning that each B-cell produces multiple duplicates of only one type of antibody which can bind to only one type

of antigen. Once activated by a helper T-cell, B-cells differentiate into memory B-cells and plasma B-cells. Memory B-cells may survive in the body for several decades to provide a stronger and quicker immune response to a future infection by the same intruder. Plasma B-cells become antibodies producing cells, supplying the bloodstream with antibodies unique to the antigen involved in the current infection [21]. B-cells are also APCs; however, B-cell ability to recognize only one type of antigens makes DCs and MΦs more efficient and more generalized APCs. Furthermore, an antigen presenting process to a helper T-cell is still necessary to activate the B-cells [22]. The adaptive immune response using the B-cells is called humoral response.

2.2. An Overview of the Artificial Immune System

AIS has recently become a diverse area of research that attempts to take inspiration from immunology for solving engineering problems. AIS paradigm has first emerged within computer science and engineering. Over the past years, several classes of AIS-based algorithms have been developed. The AIS application included robotics and control [26]-[29], anomaly detection[30]-[34], data mining [35]-[37], optimization [38]-[40], machine learning [41], [42], network and computer security [43]-[45], pattern recognition [46]-[48], and image processing [49], [50].

2.2.1. AIS for Abnormal Conditions Management

Failure or anomaly detection has been an area of significant interest in AIS application. Typically, the goal of these immunity-based approaches is to decide whether a test sample was produced by a system under normal or abnormal operation. The early work of Forrest et al.[51], [52] led to the early immunity-inspired failure detection systems. In these works, a general method for distinguishing between self (i.e. normal operation) and non-self (i.e. abnormal operation) was proposed inspired by negative selection. The approach was applied for virus detection in a computer system. The reported results indicated the possibility to use immunity-based algorithms for intrusion detection.

Kim and Bentley investigated different evolutionary stages of AIS and applied it to network intrusion detection: negative selection [53], negative selection and static clonal selection [54], and dynamic clonal selection [55]. In [54] a combination of the static clonal selection algorithm with the negative selection algorithm used to select the detector sample size and the antigen sample size to lower the false alarm rate. In [55] the authors

proposed a dynamic clonal selection approach which dynamically adapts to continuously changing system behavior by allowing changes to self-clusters and predicting new patterns of non-self; however, human confirmation (co-stimulation) is necessary, which makes the approach dependent on human interaction.

An artificial immune regulation (AIR) approach was introduced and integrated into an immune model-based fault detection scheme for fault diagnosis in.[33]. The model-based fault detection system generated residuals that contained information about the faults. The AIR approach, inspired from the biological immune regulation process and numerical clustering techniques, is used to classify the residuals into a number of distinct patterns corresponding to different faulty situations. Various disturbances and errors were found to cause residuals to become nonzero, thus interfering with detection and identification of faults.

Guzella et al. proposed an immunity inspired approach for fault detection called dynamic effector regulatory algorithm (DERA) [56]. The approach integrates immunological regulatory T cells function in the control of various aspects of the immune system and includes a mechanism for signaling between cells. DERA uses a population of regulatory and effector T-cells, combining both positive and negative detection; it also keeps track of the concentration of two cytokines in the environment. The primary concept behind DERA is that there must be an interaction between the cells in the environment before deciding if an antigen is a self or non-self. DERA possesses a memory that is represented by the two cytokine concentrations; therefore, the classification of an antigen depends on the responses against recently classified instances. The DERA approach was tested on the DAMADICS fault-detection benchmark problem, and it was able to attain considerably lower false-positives than other approaches evaluated. One drawback of the approach is that a slowly growing fault could take a long time before being detected.

Recent research efforts are examining the interaction of the innate and adaptive immune system, rather than focusing on algorithms purely inspired from one or the other alone. For instance, the dendritic cell algorithm (DCA) [57] was proposed by Greensmith et al. to mimic the functionality of the dendritic cell of the biological immune system based on Matzinger's danger theory [58]. The danger theory proposes that the biological

immune system response is initiated by a detection of the body cell damage rather than the detection of the antigen structure. Since then, several modifications were applied to the original DCA [59], [60].Pinto et al. also used the danger theory concept to design a fault detection system for telephone system [61]. Each call in this fault detection system is represented by an antigen composed of the following features: call origin, call destination, duration of calls, and a nominal attribute. Two signal levels were identified: signal 1 for perceiving the presence of the antigen and signal 2 for co-stimulation by using the non-completed call rate. Signal 2 was responsible for alarming a danger situation.

Shu and Zhao [62] presented an immunity based fault detection and diagnosis approach for diagnosing faults in the chemical processes. The proposed approach mimics the immunity vaccination. Fault samples collected from other chemical processes of the same type are used to generate “vaccines” to help construct fault reference libraries for the fault detection purpose. Results show that the vaccines generated from similar processes can successfully diagnose different types of faults. Despite the successful results, the approach fails to identify the mean of “similar” processes and fails to use the AIS as a promising fault diagnosis approach that relies on the definition of the normal operations rather than the abnormal conditions. Zhao et al.[63] presented an online fault diagnosis system for a lab-scale distillation process. The proposed approach combines the artificial neural networks (ANNs) which are used for fault detection in the steady state, and dynamic artificial immune system (DAIS) which is used for fault detection in the startup phase and then for fault identification. Both ANNS and DAIS are trained offline for the purpose of known faults detection and identification. Unknown faults can also be detected by the system, and a user feedback is allowing operators to manually input the results to re-train the framework and include the new faults.

Remarkable research efforts at West Virginia University (WVU) have been focused on immunity-inspired aircraft abnormal condition detection, identification, evaluation, and accommodation (ACDIEA). Perhinschi et al. proposed a conceptually integrated framework for detection, identification, and evaluation of actuator, sensor, engine, and structural failures/damages [64], [65]. The detection phase represents the capability to declare failure occurrence within any of the aircraft subsystems. The identification phase defines the failed subsystem element. The evaluation phase addresses the type of the failure, its magnitude, and the reassessment of the generalized flight envelope. Moncayo

et al. proposed an immunity inspired aircraft failure detection and identification scheme [66]. The proposed framework has been successfully investigated and proved to work with a broad range of aircraft subsystem failures/damages. Moncayo et al. also proposed an immunological hierarchical multi-self (HMS) strategy in which they used an integrated multiple-self approach instead of considering the one single multi-dimensional self-configuration [67]. The proposed approach improves the detection performance significantly while maintaining the multidimensionality of the identifier space manageable. Al-azzawi et al. proposed a dendritic cell inspired mechanism for aircraft abnormal condition detection, identification, and evaluation [68], [69]. In the detection phase, the DC mechanism processes the outcomes of the self/non-self within the HMS strategy into the final detection outcome. In both identification and evaluation phases, the DC mechanism converts the identification or evaluation into a pattern recognition problem in which the failed subsystem is identified or evaluated based on pre-defined (trained) identification or evaluation patterns.

2.2.2. AIS for Control

The immunity-based methodology for control problems has been approached based on different abstractions. The first concept is the biological feedback that establishes the balance between the activation and suppression of antibodies generation. The idea was first proposed by Takahashi and Yamada [70] by modeling the production of T and B cells. Wantanabe et al. [29] proposed a decentralized adaptive control mechanism for a six-legged robot. By combining the idea of B-cells and immune networks, where a B-cell is considered to be a leg in the robot and the immune network a mechanism by which legs communicate with each other, they proposed a system that can learn how to control the walking motion of the robot. Lee et al. [71] used clonal selection algorithm to tune control parameters K_P, K_I, and K_D of proportional integral derivative (PID) controller. The proposed approach was found more efficient than Ziegler–Nichols technique in terms of settling time, overshoot, and turning the yaw angle through simulation. Krishnakumar et al. [72] and Krishnakumar & Neidhoefer [73], [74] proposed an immunized computational system (ICS) that used the immune system metaphor along with computational (both hard and soft computing) techniques to attempt to reproduce the robustness and adaptability of a biological immune system. The strategy was tested on an autonomous aircraft control problem.

Moncayo et al. [75] proposed a novel adaptive flight control system designed to handle aircraft sub-systems AC as well as upset environmental conditions. The proposed control system uses a non-linear dynamic inversion control augmented with an immunity-inspired mechanism. The simulation analysis shows that the proposed control laws increase the performance of tracking pre-defined trajectories at normal and AC flight conditions. Perez et al. [76] presented a novel humoral response inspired adaptive control law designed to maintain the performance of an aircraft under the existence of AC. The controller's parameters were optimized offline for multiple sets of AC using a genetic optimization algorithm. The presented results revealed that the proposed adaptive law reduced tracking errors and improved the pilot response required to maintain control of the aircraft under AC.

The second approach is based on the assumption that the classification capabilities of the AIS can be extended and used not only to detect, identify, and evaluate, but also to provide some solution that would minimize or exclude the AC effects. Karr et al. [77] proposed an adaptive, model-following flight control system based on AIS. The control system monitors the aircraft model for any change from the predefined reference "self" and then used an optimization algorithm to optimize the controller parameter for the current conditions. A database is used to store previous optimization results for a faster solution in similar conditions. The approach was demonstrated in the simulation of a Boeing 747 aircraft in the presence of atmospheric roughness and degradations in the performance characteristics of actuators used to manipulate various control surfaces.

Research effort at WVU extended the aircraft ACDIE for solving the AC accommodation problem [78]. The main objective of the research was to investigate the possibility of extracting compensatory commands from the AIS to address the accommodation problem. The idea relies on generating artificial memory cells, which represent the self (nominal conditions) and the non-self (AC). The self and non-self memory cells consist of measurement strings over a pre-defined time window. Each string is a set of features, which capture the dynamic fingerprint of the aircraft operation at nominal and AC, values at each sample time of the flight. The accommodation process then works as follows: the collections of data over several time samples of current flight are compared to the self and non-self memory cells. Once the best match is found, a pre-

defined control command corresponding to the current conditions is extracted from the memory cells and used for control purposes.

2.2.3. AIS for Optimization

Optimization has received a great attention as a promising application for immunity inspired algorithms. Several research efforts reported favorable outcomes when comparing immunity-based algorithms against other state of the art optimization algorithms. Most of the AIS optimization algorithms are built upon the clonal selection theory. In the cell-mediated immune response, B-cells are exposed to antigens. The B-cell that bind to the antigen receives a signal from the helper T-cells to proliferate (clone) and mature into plasma cells, which divide rapidly to generate the antibodies and memory B-cells, which serves as immunity memory. The biological immune system can be viewed as having multiple and possibly overlapping and contradictory goals. It improves its own response towards particular goal(s) as the result of feedback [79]. In general, the immune response goals reflect a compromise between the strategy of the immune system and the physiologic function of the tissues at the infection site.

The majority of current publications in the immunity optimization area are based on the application of the clonal selection principle, resulting in a number of algorithms such as Clonalg algorithm [80], opt-AINET [81], and the B-Cell algorithm [82]. All of these algorithms essentially evolve solutions to problems via repeated application of cloning, mutation, and selection cycle to a population of candidate solutions (B Cells). A single antigen represents some function to be optimized, and good solutions are allowed to remain in the population, mimicking the memory cell mechanisms believed to exists in the natural immune system. Freschi and Repetto [83] provided wide-ranging analysis of opt-IA and Clonalg algorithms using a robust test-bed that includes multiple optimization problems such as max-1s, trap functions and 23 numerical optimization problems from reference [84] and find that immune algorithms are comparable to some of the most effective methods in the evolutionary algorithm literature such as fast evolutionary programming (FEP). Timmis et al. [82] compared versions of opt-AINET and the B-Cell algorithm to a variety of optimization problems of various dimensions found in the literature, and [85] applied Clonalg to a range of constrained optimization problems.

Wojdan et al. [86] presented an optimization method of a combustion process in a power boiler using immunity-inspired optimizer namely SILO. The main goal of SILO is optimization of power station's variable costs, achieved by minimization of CO and NOx emissions. The approach was shown to decrease implementation costs and adapt to new operation environments, be usable in practice, and be a good alternative to MPC controllers.

Chapter 3. AIS-Based Framework for Monitoring and Control

This chapter presents an immunity-based framework for the power plant abnormal conditions detection, identification, evaluation, and accommodation (ACDIEA) problem. It starts with the definitions of the various terms and components used in designing the framework, then introduces the general architecture of the ACM process. It should be noted that the targeted system in this work is the IGCC power plant, mainly the acid gas removal (AGR) unit, which will be later used for demonstration. Reference to this specific system will be made throughout the document. However, the framework is formulated such that it can be applied to any other complex dynamic system.

3.1. General Architecture of the AIS-Based Monitoring and Control Framework

Online ACDIEA system for chemical processes should at least meet the following requirements:

1. The online monitoring and control framework should have an integrated structure, in order to address the complexity and multi-dimensionality of the chemical process in a coherent and consistent manner.

2. The framework should be comprehensive and capable of addressing all scenarios involving all relevant subsystems and the possible, known and unknown, ACs.

3. The set of ACDIE algorithms should be fast enough to detect, identify, and evaluate the AC, ideally in one single time step. The control framework should be fast enough to provide adequate AC accommodation.

4. The system should have online-learning capability to adapt to new conditions. Due to the complexity of the chemical process, equipment may not be able to maintain an exact operation state throughout the entire equipment service life cycle.

5. The development and use of all tools and mechanisms within the framework must be straightforward, simple, and affordable.

The immunity-based power system monitoring and control process considered in this research includes the development and implementation of three principal components functionally connected in a closed loop as presented in Figure 3-1 below.

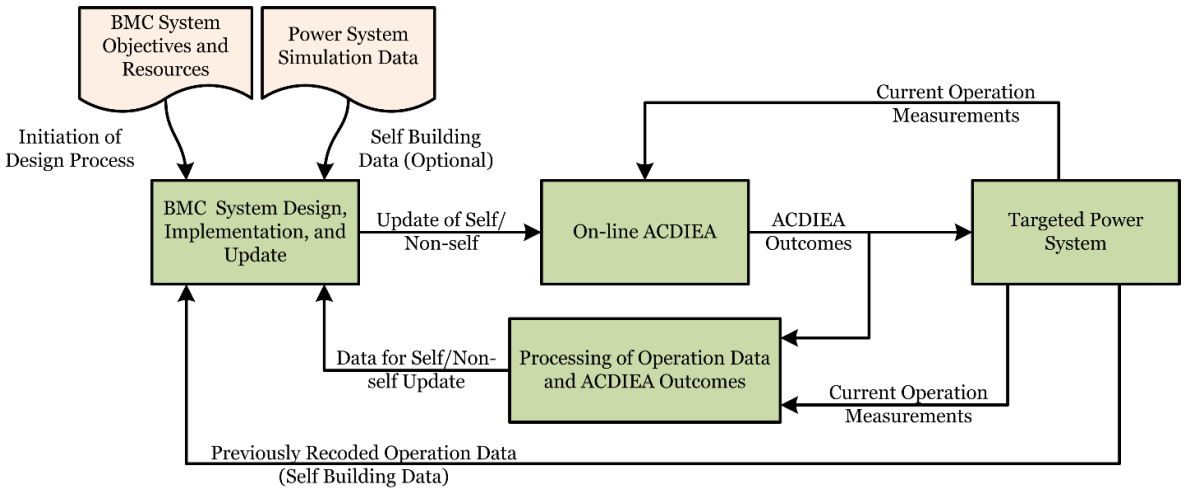

Figure 3-1. Power System Biomimetic Monitoring and Control Process

The **biomimetic monitoring and control (BMC) system design, implementation, and update** represents the initial development of an integrated and comprehensive ACDIE scheme, control system scheme (baseline and adaptive components), and evolutionary optimization modules. Then, during operation of the system it may use newly acquired data to update the AIS and the scheme.

The **on-line ACDIEA** process implies the real time operation of the ACDIEA scheme. Sets of current values of the features measured during current system operation at a certain sampling rate are compared against the self and/or non-self using various algorithms to generate the final ACDIE outcomes. These results are transferred to the supervising/operating personnel and the automatic fault tolerant control laws.

The **processing of operation data and ACDIEA outcomes** involves analyzing the false alarms and failed detections in conjunction with current measured values of the features for modifying/extending the self/non-self and improving the overall performance of future operations. This process is also envisioned to re-assess the operational conditions, re-optimize parameters, and/or trigger switching between different operational modes.

The power system BMC process is accomplished through the development and implementation of three principal components as illustrated in Figure 3-2:

- AIS-based ACDIE
- Control System: Baseline + Adaptive Component
- Evolutionary Optimization Module

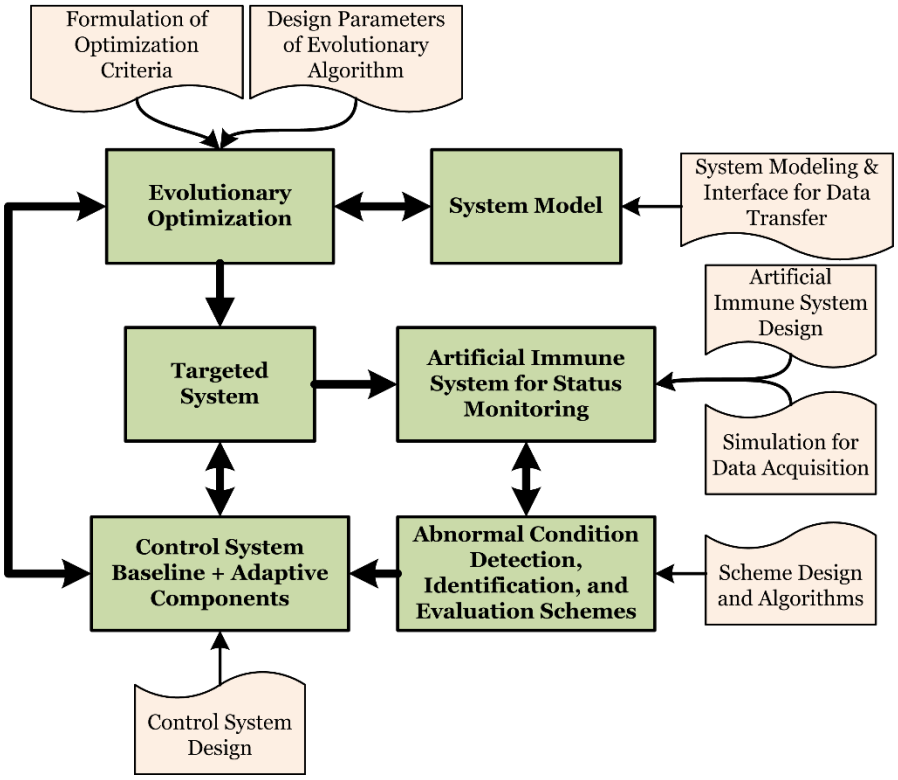

Figure 3-2. General Framework Architecture

The ACDIE scheme along with the baseline controller mimics the innate immune system as the first line of defense against external intrusion. The design of the ACDIE scheme requires a clear definition of the targeted system subsystems and components that are targeted, the types of AC (including known and unknown ones), the AC severity scale, the operational envelope variables, and the nature and level of passive and active accommodation. The development of the immunity-based ACDIE scheme also requires the availability of large amounts of measured data that must be pre-processed for self/non-self generation. The design of the baseline controller requires a clear definition of the control system objectives and the proper selection of the control system algorithms and parameters. The baseline control parameters are optimized using the evolutionary algorithms to ensure optimum normal operation. Similarly to the baseline control system, the design of the adaptive controller requires the definition of the system objectives, the control system algorithm, and parameters. The adaptive control parameters are optimized using the evolutionary algorithms to ensure minimal interference with the baseline controller under normal conditions, while ensuring needed accommodation under different AC. The design of the evolutionary optimization algorithms requires an accurate system model, optimization objectives, and definition of targeted parameters.

3.2. AIS – Definitions and Concepts

The ***feature variables*** or simply ***features*** are the set of variables φ_i that completely define the targeted system and are expected to capture the fingerprints of all considered AC, regarding occurrence, presence, and severity. They can be system states, inputs, control variables, parameters, estimated values, etc. The set of all features $\mathcal{F}$:

$$\mathcal{F} = \{\varphi_i \,|i = 1,2, \dots, N\} \tag{3-1}$$

defines a ***feature point*** as a set of simultaneous values of all features φ_i that can be obtained through measurements or simulation, at normal or abnormal conditions. The set of all possible feature points defines an N-dimensional hyperspace U, which will be referred to as the ***universe***. The ***self*** , S , is defined as the set of all possible feature points under normal operation conditions. Therefore, all other points in the universe are considered ***non-self*** Ŝ.

Features selection is one of the most critical steps in the design of the immunity based ACM system. The selected features must be relevant to all four components of the ACDIEA process. Their number and nature depend on the targeted system and its components, the types of the AC, the severity scales of the AC, and the nature and level of the passive and active accommodation.

3.3. ACDIE Problem Formulation

Let the targeted ***system*** be composed of a number N_S of subsystems, possibly nested, such that each subsystem l may be composed of N_{cl} sub-subsystems or components. Since the targeted system in this work is a power plant, the subsystems include all the power plant hardware, measurement and control devices, such as actuators, sensors, automation devices, and others. Human operators and the environment may also be considered part of the targeted system. In general, within a subsystem, a component may interact both ways (input/output) with one or more other components creating internal loops. Similarly, subsystems may interact with one or more other subsystems. This structure is illustrated in Figure 3-3.

The system input refers to those variables that are generated outside the system (for example, they can be dynamically varied by the user for control purposes), while the system parameters are considered internal features, constant during system operation,

which characterize the system. The system output consists of variables produced by the system (controlled variables and others). These variables may be used in the calculation of the optimization criteria, for control and/or monitoring purposes.

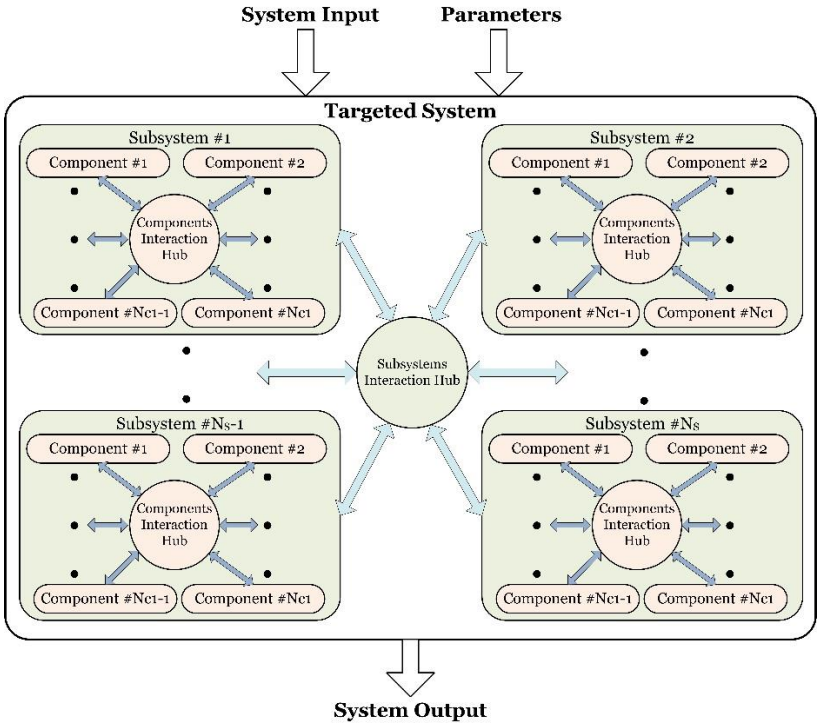

Figure 3-3. Targeted System and Its Components

The term ***abnormal conditions*** **(AC)** refers to situations that are outside the general design framework and require specific attention and/or action for system performance and safety purposes. AC include hardware faults and failures, exceedance of nominal operational ranges, human operator related abnormal situations, operational upset conditions, and extreme environmental conditions.

The AC ***detection*** process is the recognition of the presence of an abnormal condition in at least one of the subsystems or components. The detection outcome, Det, is binary and can be express as:

$$Det = \begin{cases} 0 & no\ AC \\ 1 & AC\ present \end{cases} \tag{3-2}$$

The AC ***identification*** or ***isolation*** process determines which subsystem has failed. Depending on the complexity of the targeted system, the AC identification can be performed in several phases. For example, a first identification phase could determine

that a certain subsystem such as an absorber or turbine has been affected by an AC. A second phase could determine which specific component has failed, such as a valve or a pressure sensor. The outcome of the subsystem identification process, Idt, can be formulated as an N_s-dimensional vector with binary components or as a set of integers labeling only the failed subsystems. The outcome of the identification process is formulated in equation (3-3). An extension to address component isolation can be easily produced.

$$Idt = [id_1,\ id_2, \dots,\ id_{N_s}], id_k = \begin{cases} 0 & subsystm\ k\ is\ AC\ free \\ 1 & subsystem\ k\ is\ under\ AC \end{cases} \quad (3\text{-}3)$$

The AC ***evaluation*** process addresses three different aspects. The AC ***qualitative evaluation*** (QE) is the determination of the failure type. The outcome of the qualitative evaluation, EvQ, is an integer labeling the type of failure out of a list of F_{NK} failure types associated with each component or subsystem $k = 1, 2, \dots, N_s$, such that:

$$EvQ = fi:\ fi \in \{1, 2, \dots, F_{NK}\} \quad (3\text{-}4)$$

The AC ***direct quantitative*** evaluation step is the estimation of the failure severity. The outcome of the direct evaluation ($EvDQ$), can be represented in two different ways. Using a numerical approach, $EvDQ$ can take a value between 0 and 1, where 0 represents no failure, and 1 represents the highest severity of the failure. An example for the numerical approach is:

$$EvDQ = 0.36 \quad (3\text{-}5)$$

On the other hand, $EvDQ$ can take pre-defined interval range values to represent the failure severity. The pre-defined ranges can be represented as discreet values, such that:

$$EvDQ \in \{low, medium, high\} \quad (3\text{-}6)$$

where *low* may represent a severity value between 0 and 0.33, *medium* represent a severity value between 0.34 and 0.66, and *high* represent a severity value higher than 0.67.

The AC ***indirect quantitative*** evaluation is the assessment of the AC effect on the power plant performance and constraints. The outcome of the indirect quantitative

evaluation ($EvIQ$)process typically represents a set of new ranges at post failure conditions of the variables that define the system operational envelope.

3.4. Accommodation Problem Formulation

The AC ***accommodation*** process can take two forms: ***passive accommodation*** through delivering ACDIE outcomes and other warnings and information to the supervising personnel and ***active accommodation*** through direct compensation as an integral part of the system control laws. A simplified block of AC active accommodation is shown in Figure 3-4.

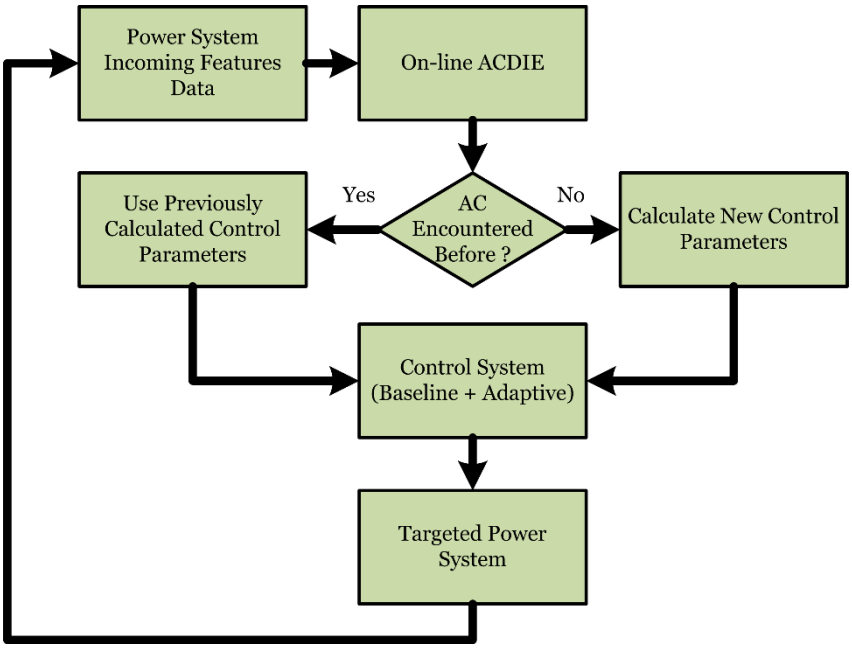

Figure 3-4. AC Active Accommodation

In order to accomplish the active accommodation problem successfully, the three components of the ACDIE must deliver accurate and timely outcomes. In other words, the existence of an AC, the failed subsystem failed, and AC type and severity, must be provided with high reliability rate. This is needed for the fault-tolerant control system to provide a timely accommodation. The ultimate step of the AC active accommodation is accommodating the control system to the existent AC. It represents the actual reaction of the control system to compensate the AC by generating appropriate control commands. The first scenario assumes that substantial information about the AC and its compensation are available and stored within the framework. The second one involves an unknown AC that requires a specific new compensation. In this case, the control

command may be calculated using online learning capable adaptive control or using an optimization algorithm. The newly calculated control commands are then saved for faster future accommodation.

3.5. Evolutionary Optimization Problem Formulation

Within the ***evolutionary algorithms (EA)***, an individual is a potential solution to the optimization problem, which is, in this context, a single set of control system parameters or gains. An individual I can be represented as:

$$I = [g_q], q = 1,2, \dots, N_g \tag{3-7}$$

where g_q represent the q^{th} parameter value of the total number of parameters N_g.

The optimization problem solution space is defined as the set of all possible individuals. The size of the solution space is calculated using the combination formula. In other words, the size of the solution space SS equals to:

$$SS = \prod_{q=1}^{N_g} gv_q \tag{3-8}$$

where gv_q is the number of all possible values for g_q. The value of gv_q can be calculated by defining minimum value g_{qmin}, maximum value g_{qmax}, and a uniform resolution g_{qres} to each parameter g_q. The resolution g_{qres} represents the smallest difference between two parameters g_q. gv_qthen defined as:

$$gv_q = \frac{|g_{qmax} - g_{qmin}|}{g_{qres}} + 1 \tag{3-9}$$

The fitness function is used by the EA to evaluate the performance of a given individual (i.e. solution) with respect to the optimization problem. The fitness function rewards desired performance and penalizes undesired performance or constraints violations. The fitness function FF relies on the establishment of a set of N_p performance criteria pe and associated numerical metrics to form a performance vector PE such that:

$$PE = [pe_1\ pe_2\ \dots\ pe_{N_p}] \tag{3-10}$$

The normalized performance vector $\overline{PE}$ is a scaled version of the performance vector PE. The values of PE vector may be scaled based on user-specified lower and upper limits to take values within a range between zero and one by using linear function, trapezoidal function, or even fuzzy logic such that:

$$\overline{PE} = \left[\overline{pe}_1\ \overline{pe}_2 \dots \overline{pe}_{N_p}\right] \tag{3-11}$$

where, ideally, the best individual is expected to have $\overline{PE}$ as a vector of ones.

A set of N_p weights PW are established to reflect the relative importance of the evaluation criteria and/or to accelerate improvement along specific directions of the overall fitness. The set of weights is defined as:

$$PW = \left[pw_1\ pw_2\ \dots\ pw_{N_p}\right] \tag{3-12}$$

The overall fitness FF of a potential solution is then defined as the weighted average of all elementary performance evaluations:

$$FF = PW^T.\overline{PE} \tag{3-13}$$

The EA is ran until FF converges to 1 or a pre-defined number of individuals have been evaluated. The EA is explained in details in Chapter 7.

Chapter 4. The Partition of the Universe Approach

Within the AIS paradigm, a critical element is the representation of the system under normal conditions (*self*) and under AC (*non-self*). The typical approach relies on a negative selection-type of technique [87], [88] consisting of clustering self-data and then covering the non-self with similar clusters, which are viewed as detectors. In this chapter, an alternative to the data clustering approach (DCA) is introduced that avoids the computational burden of clustering and covering of the non-self, while allowing using the entire multi-dimensional self.

4.1. Self / Non-self Discrimination

The discrimination between self and non-self is the fundamental concept based on which the AIS paradigm is constructed. Therefore, generating an AIS for a technological system is centered around the definition of the regions in the feature hyperspace that corresponds to the normal operation (self) and regions that correspond to abnormal operation (non-self). Generating the self, S, requires collecting significant amounts of measured feature values at normal conditions, ideally covering the entire operational envelope. These measurements can be gathered from the actual functioning plant, from simulation, or a combination of the two.

4.2. Preliminary Data Preparation

Once the normal operation of the system is defined and the operation data is collected, the quality of the normal operation plant data must be verified to prevent corruption from hardware malfunction, operational constraint violation, and other perturbations. Next, the data must be organized according to the structure of the AIS, which can consist of a single multi-dimensional entity or multiple lower dimensional projections of the feature hyperspace, within the HMS strategy [67]. Data is then normalized between 0 and 1, using normalization factors for each feature determined as the span of the feature data plus a percentage margin. Note that if data is collected in multiple sessions, consistency of the normalization factors must be ensured. Duplicate points of the normalized data are eliminated to reduce the size of storage memory and computing resources. Figure 4-1 summarizes the preliminary data processing for AIS generation.

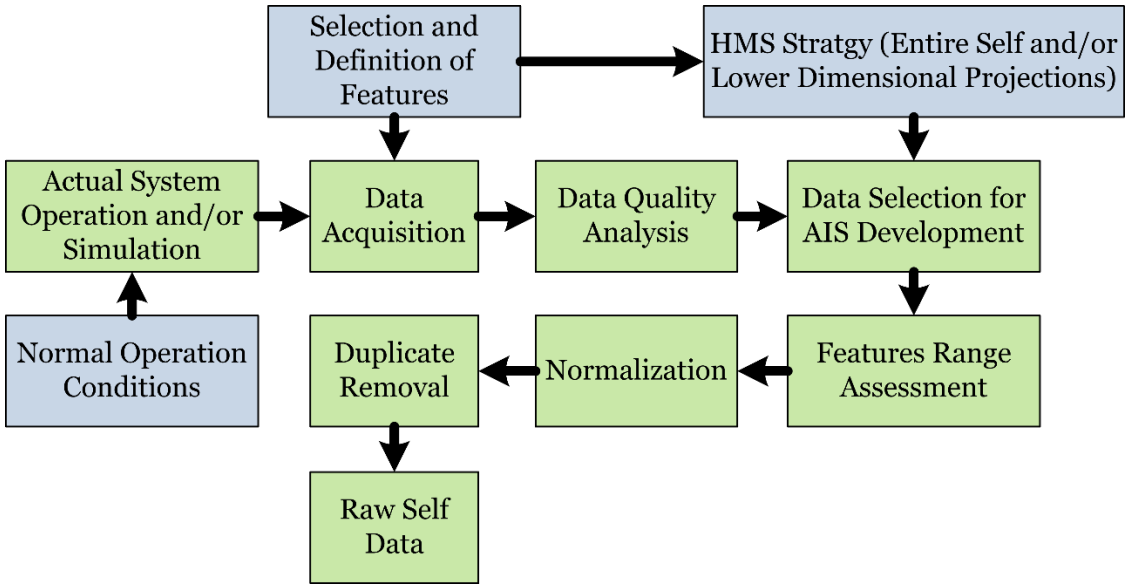

Figure 4-1. Data Preparation for Self Generation

4.3. The Self Generation Using Data Clustering Approach

Clustering self data requires defining several parameters depending on the algorithm used. In general, the clusters' shape, size, and number are essential. When using the clustering approach, the N-dimensional feature points are clustered using algorithms such as K-means. K –means [89] is a popular approach used to partition any given set of data into k clusters, such that each point from the data set will belong to the nearest cluster. The algorithm starts by setting k number of initial clusters' centroids called seed-points. K-means then calculate the distances between the input data points and the centroids and assign each point to the nearest centroid. The new clusters' centroids are then calculated by using the data points that belongs to each cluster. The process then repeated until the clusters' centroid have converged. Figure 4-2 summarizes the K-means algorithm.

As a result of the K-means clustering, the self is represented as a set of geometrical hyper-bodies, such as hyper-spheres, hyper- rectangles, or hyper-ellipsoids [90]. An optimization process is typically used to minimize the empty space and the overlapping between self-clusters, and minimize the number of clusters. Once the self is generated, similar hyper-bodies are used to cover all the non-self areas, which are referred to as antibodies or detectors. When generating the non-self, the following optimization process is typically applied to minimize self-clusters and detectors overlapping, reduce overlapping among the detectors, and minimize the number of detectors. Previous studies

in the areas of AIS chose circles as a shape and the K-means algorithm for clustering, resulting in 2-dimensional self projections, such as illustrated as in Figure 4-3.

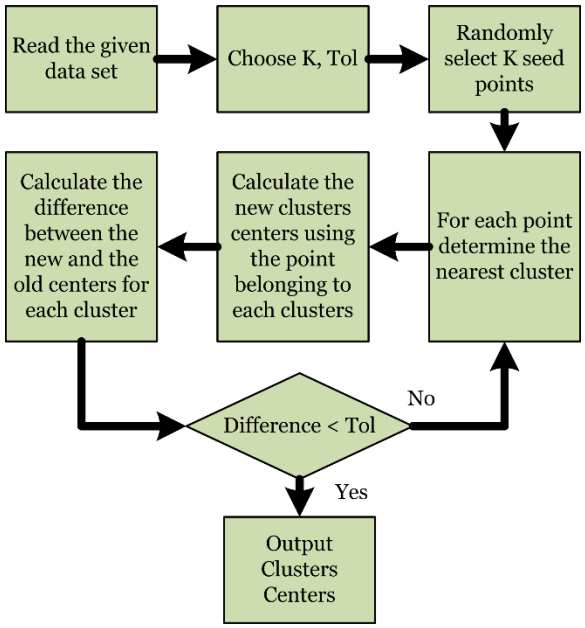

Figure 4-2. K-means Algorithm for Data Clustering

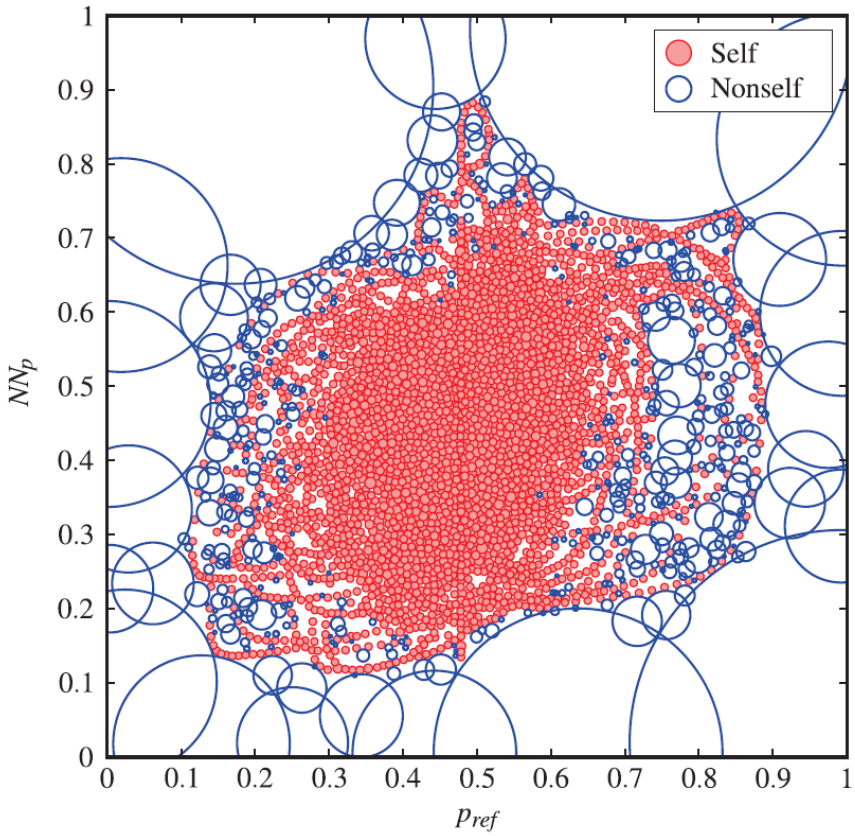

Figure 4-3. Sample 2-D Self Projection Generated Using K-means [91]

The processes of clustering self-data and covering the non-self are computationally intensive. The average time needed to produce one two-dimensional self reported by previous dissertation [91] was 4450 sec using clusters set union method. In that research, uniform 2- dimensional projections were used to represent a technical system self. In other words, the estimated time needed to generate the total number of 469 projections for a system with 32 features in the previous research was (2087050 sec) or approximately 24 days. The total number of the uniform projections UP is calculated based on the number of features N and the projection dimensionality D using the following combination formula:

$$UP = \frac{N!}{D!\,(N-D)!} \tag{4-1}$$

It is clear that the number of uniform projections grows exponentially with the increase in the number of features. For example, a system with 100 features, the number of 2-dimensional uniform projections needed is 4950 projections, which translate to 253 days of continuously running DCA. The DCA becomes impractical to use for a system with a high number of features, and an alternative approach must be adopted.

4.4. The Self Generation Using Partition of Universe Approach

When using the partition of the universe approach (PUA) [92] , the universe is divided into partition clusters with predefined shape and resolution. The raw self data points are then tested against partition clusters, and self clusters are identified and labeled. As a result, the self is represented by strings of integers identifying the self partition clusters. The detection can be performed using the entire high dimensional self through a positive selection-type of logic without significant computational issues. Since the partition of the universe is already defined, the non-self results implicitly.

After the preliminary data processing, the self could be generated by using the PUA. Let the self be defined by a set of features $\mathcal{F}$ such that:

$$\mathcal{F} = \{\varphi_i |\, i = 1,2,\dots,N\} \tag{4-2}$$

where N is the number of features in a selected subsystem or component. Firstly, for each normalized feature (φ_i) a partition resolution must be chosen based on a proper balance between computational effort and accuracy in capturing the system dynamics. The

partition resolution consists of an integer π_i representing the number of intervals in which the unit side i of the universe hyper- rectangles is divided. The resolution set is thus defined as:

$$\Pi = \{\pi_i \mid i = 1,2, \dots, N\} \tag{4-3}$$

Uniform partition, which will be referred to as the *uniform universe grid,* could be used, where the resolution along all axes is the same. In this case, the universe will be partitioned into a set of π^N N-dimensional hyper- rectangles, where Π:

$$\Pi = \{\pi, \pi, \dots, \pi\}, length\ (\Pi) = N \tag{4-4}$$

If the sampling rate of the measured data is determined properly, such that it allows for capturing the dynamics of the system under normal and abnormal conditions, then the partition size can be selected such that:

$$\pi_i = round\ (\frac{1}{mean(\Delta\sigma_i)}) \tag{4-5}$$

where $\Delta\sigma_i$ is the normalized difference between two consecutive measured samples of feature φ_i.

Generating the self with PUA consists of comparing the self-data points against the universe grid, and whenever a feature point falls inside a partition, that partition is labeled as self. The value of each feature is positioned within the partition of the corresponding axis, and the partition label is recorded. Let us assume that a raw self-data point is represented as:

$$P_k = \{\varphi_1(k), \varphi_2(k), \dots, \varphi_N(k)\} \tag{4-6}$$

After normalization, this point is represented as:

$$\bar{P}_k = \{\bar{\varphi}_1(k), \bar{\varphi}_2(k), \dots, \bar{\varphi}_N(k)\} : \ \bar{\varphi}_i(k) \ \in [0,1] \tag{4-7}$$

Each raw self-data point generates a self-partition cluster C_k such that:

$$C_k = \{p_{k1}, p_{k2}, \dots, p_{kN}\} : \ p_{ki} > 0 \ and \ p_{ki} \ \in \mathbb{Z} \tag{4-8}$$

and

$$\frac{p_{ki}-1}{\pi_i} \leq \bar{\varphi}_i(k) < \frac{p_{ki}}{\pi_i} \tag{4-9}$$

In case of hyper- rectangles, the self partition cluster C_k becomes:

$$C_k = \left\{ \frac{\bar{\varphi}_1(k)}{\pi_1} + 1, \frac{\bar{\varphi}_2(k)}{\pi_2} + 1, \dots, \frac{\bar{\varphi}_N(k)}{\pi_N} + 1 \right\} \tag{4-10}$$

In case of 2-dimensional uniform hexagons, the self partition cluster C_k becomes:

$$C_k = \left[\begin{bmatrix} \frac{\sqrt{3}}{3} & -\frac{1}{3} \\ 0 & \frac{2}{3} \end{bmatrix} * \begin{bmatrix} \bar{\varphi}_1(k) \\ \bar{\varphi}_2(k) \end{bmatrix} \div \pi \right]^T \tag{4-11}$$

The PUA algorithm is presented in Figure 4-4. Note that any possible duplication of self feature points is removed by completing the definition of the self process. Eventually, the self will consist of N_c hyper-rectangles similar to the self clusters obtained using DCA, but with much less computational effort. Each self hyper-rectangle $j, j = 1,2, \dots, N_c$ is now represented by a string of N elements, with each element p_{ji} being the partition label, an integer between 1 and $\frac{1}{\pi} + 1$. Self is an array of size $N_c \times N$ as illustrated in Figure 4-5. This is equivalent to a database of biological markers of the self that are used in the process of generating antibodies through negative selection. However, they will be used as detectors later in a process which is a rather positive selection-type of approach.

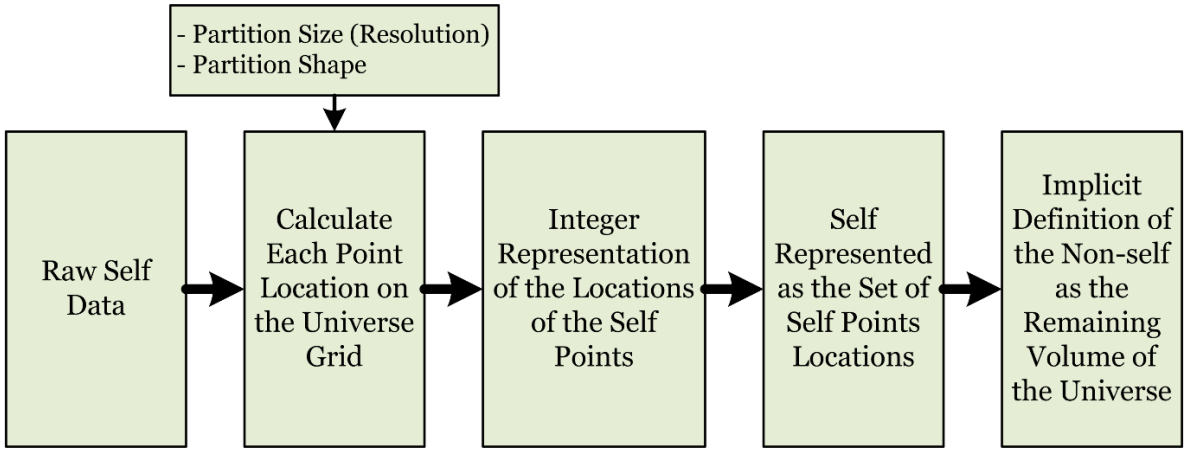

Figure 4-4. The Partition of Universe Approach Algorithm

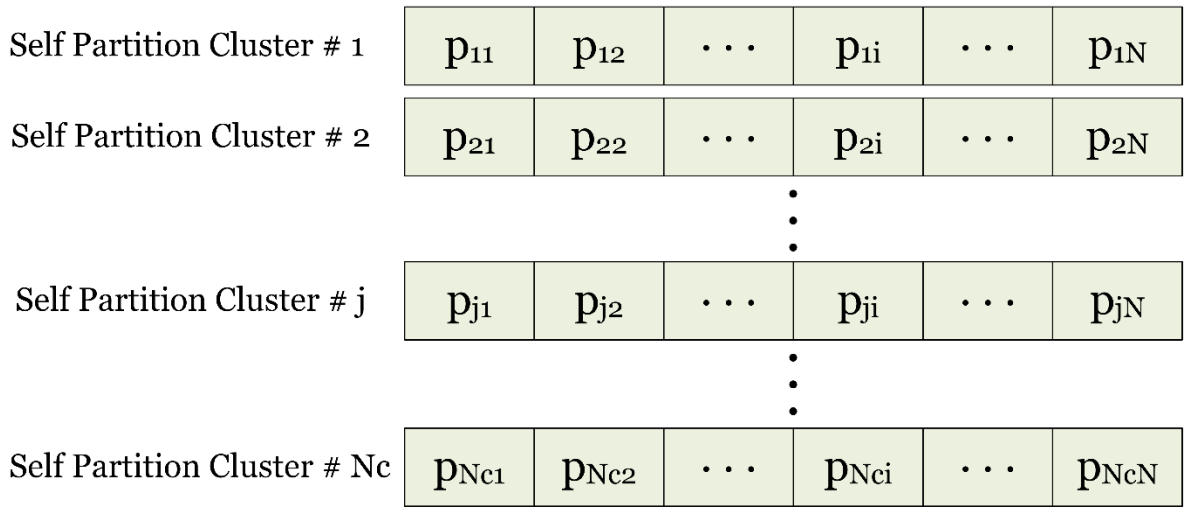

Figure 4-5. Self Representation for PUA

Within the PUA, the generation of the non-self is implicit [93], since the universe grid has a finite number of non-overlapping clusters already covering the non-self. For high resolution partitions, the size of the non-self may become impractical and require additional processing such as reducing the usable domain of non-self and/or varying partition resolution. Figure 4-6 and Figure 4-7 present sample 2-dimensional self generated using PUA with different shape partitions.

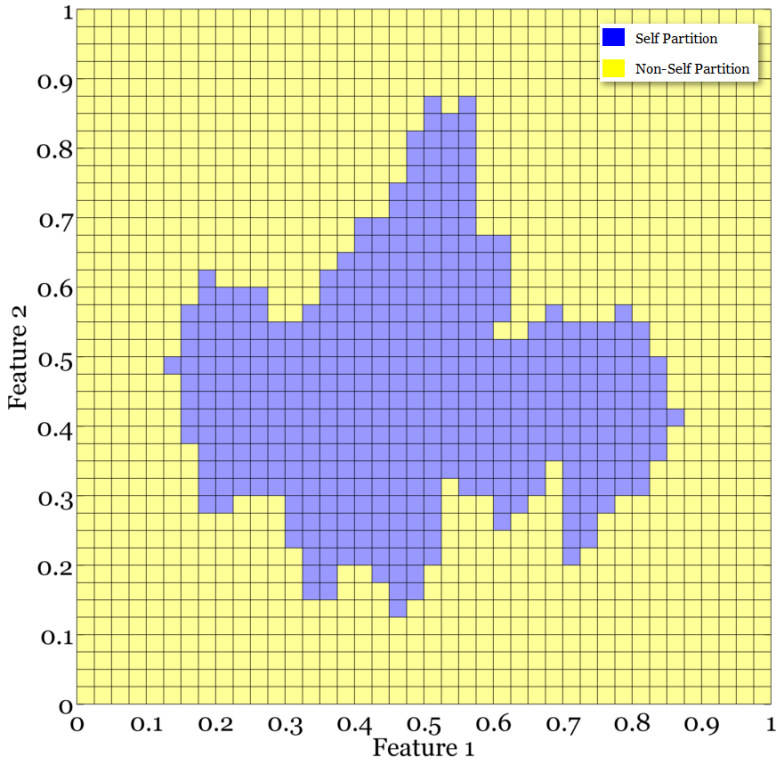

Figure 4-6. Sample 2-D Self Generated Using PUA (Uniform Square Grid)

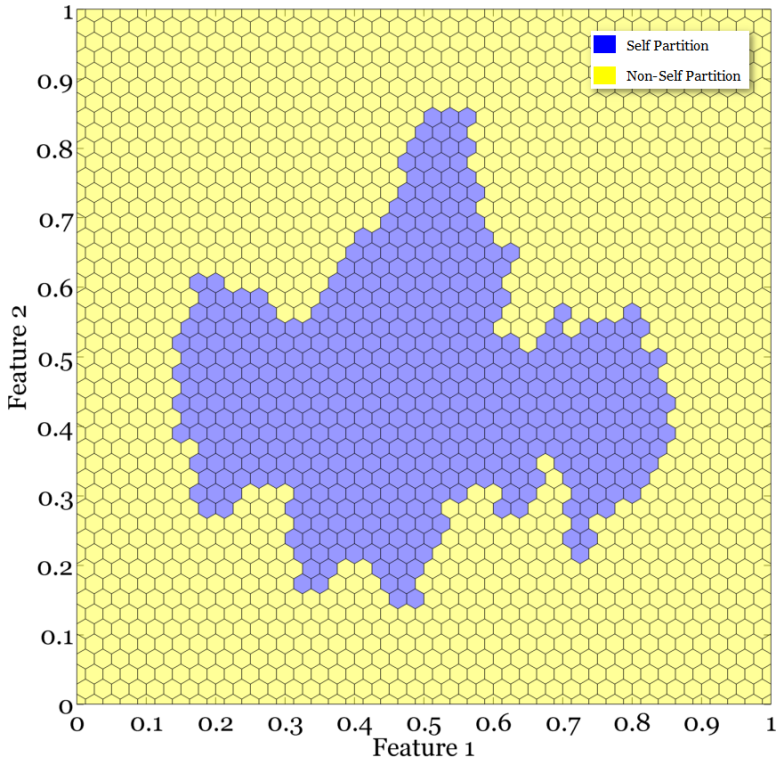

Figure 4-7. Sample 2-D Self Generated Using PUA (Uniform Hexagon Grid)

4.5. PUA vs. DCA

This section is intended to compare the proposed PUA and DCA in the context of building the AIS self/non-self in terms of algorithm parameters and computational issues.

4.5.1. Algorithm Parameters

The PUA algorithm requires the selection of the universe partition shape and size. The two possible selections in which the entire universe is covered without overlapping between partitions are hyper-rectangles and hyper-hexagons. Hyper-rectangles are the generalization of a rectangle in higher dimensions, while a hyper-hexagon represents a set of uniformly distributed points in the N-dimensional space, such that each point has a distance equal to twice as the size of the hexagon from the neighborhood points. In this research, only hyper-rectangles have been considered. Further studies are needed to determine the effect of the shape selection on the final generated self and therefore the ACDIE performance. The selection of the partition size for the PUA could be calculated using Eq. (4-5) or selected heuristically. It is critical to select the right partition size. Selecting a smaller partition size could cause discontinuities in self and therefore causing

a higher rate of false alarms. On the other hand, selecting a larger partition size could include areas from the non-self region into the self causing missed detections.

The DCA algorithm requires the selection of the cluster numbers, shapes, and sizes. It is very critical to select the right number of clusters. Selecting a higher number of clusters could yield slower algorithm and might cause discontinuities in self and, as a consequence higher rate of false alarms. Selecting a lower number of clusters may result in covering areas from the non-self, causing missed detections. DCA allows for more flexibility in term of shape selection, since spherical or ellipsoidal shapes may also be considered; however, overlapping between detectors or uncovered self/non-self areas will occur.

At first sight, it seems that the selection of partitions size for the PUA is an equivalent issue to selecting the number of clusters using DCA. However, the partition size for each feature could be either calculated using Eq. (4-5) or fine-tuned using different partitions size and then analyzing the continuity of the resulted selves. The fine-tuning process could be easily achieved because of the PUA speed and the possibility to mathematically test selves' continuity. On the other hand, there is no easy way to calculate the required number of clusters and fine tuning could be time-consuming due to DCA algorithm computational time requirement and the difficulty to mathematically test selves' continuity.

4.5.2. Computational Issues

The PUA algorithm is not iterative, which means that each point is only processed once to determine its corresponding partition. The PUA processing time depends only on the size of raw self data point count. The PUA does not require any pre -duplicate data removal to accelerate its convergence. In fact, the approach implicitly removes duplicate points much faster than typical duplicates point removal approaches. The small processing time of the PUA allows the use of high-dimensional selves without posing any significant computational issues.

The DCA approach, on the contrary, uses a slow iterative process. The DCA processing time depends on the number of raw self data points counts, selected number of clusters, and the clusters dimensionality. Duplicate data points removal is essential to enhance DCA convergence time. In this process, the distance between each two data

points are compared to a pre-define threshold, and one of the data points is deleted when the distance is smaller than the threshold. Using a high number of clusters and higher dimensional clusters could exponentially slow the algorithm, which forces to use a limited number of clusters at lower dimensions.

Table 4-1 presents time needed for each of the methods to process three sets of 2-dimensional data points into self. The computational time is calculated for the algorithm only, it excludes the normalization process needed for both algorithm, and the duplicated data removal needed for the DCA.

Table 4-1. Sample Computational Time for PUA and DCA

Computer Specifications		**Intel Core i7 @2.93, 16 GB RAM**	
Operating System		**Windows 7 Enterprise 64-bit**	
Development Environment		**Matlab® 2014a**	
Clustering Method		**PUA**	**DCA (K-means)**
Computational Time (seconds)	rand(6000,2)	0.0043	3.1763
	rand(60000,2)	0.0211	55.9379
	rand(600000,2)	0.1713	1056.222

Another aspect of the computational time comparison between PUA and DCA is the discrimination time. The discrimination time is the time needed to detect whether a given feature point belongs to the self or the non-self. It is worth mentioning that each coming feature point should be normalized using the exact ranges used to normalize the raw self data before any testing.

The PUA is associated with the positive selection approach. In this approach, the normalized feature point is first converted to the corresponding partition. Then, the partition is checked to determine whether it belongs to the self. If the partition is determined to belong to self, then the point is considered a self-point; otherwise, the point is considered a non-self point. The discrimination time using the PUA and the positive selection is very small, and a minimum number of partitions are checked in the process.

The DCA could be associated with both positive of negative selection approach. In the positive selection approach, the self clusters are iterated to check whether the given point belongs to one of them. If the point is found to belong to a cluster, the process is stopped, and the point is judged to be a self-point; otherwise, the point is considered to belong to the non-self. In the negative selection approach, the non-self detectors are iterated to check whether the given point belongs to one of them. If the point is found to

belong to a detector, the process is stopped, and the point is said to be a non-self point; otherwise, the point is said to belong to the self. It is clear that the discrimination time using DCA depends on the number of clusters and detectors in the self/non-self, respectively.

The last aspect of the comparison is the amount of memory needed to store the self/non-self data using PUA and DCA, assuming both approaches are used to build selves of the same dimensionality. PUA yielded number of partitions depends on the selected resolution, while DCA yields the selected clusters numbers. At first, DCA seems to offer more flexibility regarding the memory needed to store the self/non-self data when a limited-size memory is available; however, because of the use of a pre-defined universe in PUA, memory encoding (i.e. conversion to binary) could be used to compress the self data.

4.5.3. Comparison Summary

In summary, the proposed PUA is superior when compared to DCA in the context of building the self/non-self. The PUA parameters are more intuitive to the designer; the approach is much faster than DCA; does not require non-self building, provides faster self/non-self discrimination results, and has better potential to generate a self for limited memory applications.

Chapter 5. AIS-based Abnormal Condition Detection, Identification, and Evaluation

Detecting the presence of an AC, identifying the most affected subsystem, and evaluating its severity are critical steps in the ACM process. A reliable and fast ACDIE scheme is required, such that the AC accommodation schemes can provide timely and accurate compensation. This chapter introduces the basic concept of self/non-self discrimination and discusses its general applicability to the ACDIE. Due to the computational issues involved in generating the self and non-self (i.e. collecting all possible normal operation data, selecting the right partition size, noisy data etc.) a perfect definition of the self/non-self is hard to achieve in practice. Using solely direct self/non-self discrimination at each instant may lead to false alarms and/or missed detections; however, previous investigations [91], [94] have shown that false alarms and missed detections can be reduced by properly processing the instantaneous outcomes of the self/non-self discrimination using an artificial dendritic cell algorithm.

5.1. Self / Non-self Discrimination

Using the PUA, the discrimination outcome is obtained by first locating the feature point on the universe grid using each feature defined resolution.

Assume that the current feature point value at time sample t is $\mathcal{F}_t$ such that:

$$\mathcal{F}_t = \{\varphi_{it} | \; i = 1,2, \dots, N\} \tag{5-1}$$

where φ_{it} is the feature i values at time sample t. The normalized current feature point $\overline{\mathcal{F}}_t$ is the current feature point $\mathcal{F}_t$ normalized using each feature span value calculated and used in the self building phase:

$$\overline{\mathcal{F}}_t = \{\overline{\varphi}_{it} | \; i = 1,2, \dots, N\} \; | \; \overline{\varphi}_{it} \in [0,1] \tag{5-2}$$

The feature point location on the universe grid vector $\mathcal{F}P_t$ can be expressed as:

$$\mathcal{F}P_t = \; \{fp_{it} | \; i = 1,2, \dots, N\} \tag{5-3}$$

Note that the feature point location on the hyper-rectangles universe grid $\mathcal{F}P_t$ is calculated using the partition resolution set defined in Eq. (4-3) and selected in the self building phase, such that:

$$\mathcal{F}P_t = \left\{ \frac{\overline{\varphi}_{it}}{\pi_i} + 1 |\ i = 1,2,\dots,N \right\} \tag{5-4}$$

where π_i is feature i partition resolution selected in self building phase. On the other hand, the feature point location on a uniform 2-dimensional hexagon universe grid can be calculated using:

$$\mathcal{F}P_t = \left[\begin{bmatrix} \frac{\sqrt{3}}{3} & -\frac{1}{3} \\ 0 & \frac{2}{3} \end{bmatrix} * \begin{bmatrix} \bar{\varphi}_{1t} \\ \bar{\varphi}_{2t} \end{bmatrix} \div \pi \right]^T \tag{5-5}$$

Using the HMS strategy, the targeted system self S may consist of a number N_p of lower-dimensional self projections. These self projections may have homogenous or non-homogeneous dimensions, such that:

$$S = \{S_l |\ l = 1,2,\dots,N_p\} \tag{5-6}$$

A self projections-features mapping matrix (SF) can be constructed in which the membership of each feature is mapped to the corresponding self projection such that:

$$SF = \begin{matrix} S_1 & S_2 & \dots & S_{N_p} \end{matrix} \begin{matrix} \\ \\ \\ \\ \end{matrix} \begin{bmatrix} \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1N_p} \\ \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2N_p} \\ \vdots & & \ddots & \vdots \\ \alpha_{N1} & \alpha_{N2} & \cdots & \alpha_{NN_p} \end{bmatrix} \begin{matrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_N \end{matrix} \tag{5-7}$$

where:

$$\alpha_{il} = \begin{cases} 1 & \varphi_i \in S_l \\ 0 & \varphi_i \notin S_l \end{cases} \tag{5-8}$$

Note that the dimension of the l^{th} self projection represents the number of features used to build that projection; therefore, the dimension of l^{th} self projection is equal to the sum of the l^{th} column elements of the SF matrix.

To obtain the self/non-self discrimination outcome for each self projection, the feature point location on the universe grid $\mathcal{F}P_t$ is reorganized into N_p vectors. Each vector $\mathcal{F}P_{tl}$ has the elements of the feature that belongs to the l^{th} self projection such that:

$$\mathcal{F}P_{tl} = \{fp_{it} | \forall \alpha_{il} = 1\} \quad (5\text{-}9)$$

note that the order of fp_{it} elements should match the order of the features in the corresponding sub-self and the size of $\mathcal{F}P_{tl}$ should match the l^{th}self projection size.

Finally, the self/non-self discrimination outcome Q_{tl} outcome for each self projection l is then calculated using the following formula:

$$Q_{tl} = \begin{cases} 0 & if\ \mathcal{F}P_{tl} \in S_l \\ 1 & if\ \mathcal{F}P_{tl} \notin S_l \end{cases} \quad (5\text{-}10)$$

and the complementary self/non-self discrimination outcome $\overline{Q}_{tl}$ outcome for each self projection l is calculated using the following formula:

$$\overline{Q}_{tl} = \begin{cases} 0 & if\ \mathcal{F}P_{tl} \notin S_l \\ 1 & if\ \mathcal{F}P_{tl} \in S_l \end{cases} \quad (5\text{-}11)$$

If the self and non-self were perfectly defined, the discrimination results would provide accurate detection outcomes each time step. However, a perfect definition of the self/non-self is unattainable, which may cause false alarms and/or missed detections. Therefore, the self/non-self discrimination should be processed in a certain way before providing the detection outcome to eliminate false alarms, while maintaining a high detection rate. The strategy of attempting the elimination of false alarms at the possible expense of detection rate, instead of just reducing the false alarms, is justified by the fact that the system typically operates most of the time at normal conditions and even low number of false alarms may impact performance significantly. This can be achieved by incorporating an additional detection logic, such as the artificial DC, to provide the detection outcome as a function of current and past discrimination outcomes within the HMS strategy.

5.2. The Artificial Dendritic Cell Mechanism

The artificial DC mechanism presented in this section is a modified version of the one proposed in [94]. The algorithm is a novel computational approach inspired by the functionality of the biological DCs and their role in adaptive immune system activation. The main modifications to the original algorithm are:

1. Introducing the partition tracking matrix to address the evaluation problem. In the previous version of the DC, the evaluation of AC severity was based on the pattern of the triggered/non-triggered self projections. Using the PUA, the evaluation process could be performed by tracking the subsystem feature movement in the non-self region. The new approach uses fewer features and therefore provides outcomes much faster.

2. Updating the DC $IL10$ and $IL12$ functions. In the previous artificial DC algorithm, the self projections are assumed to be homogenous and 2-dimensional. Using the PUA allowed the use of higher and non-homogeneous self projections; therefore, $IL10$ and $IL12$ functions are updated to address more general self structures.

3. Introducing a training-free AC identification. In the previous artificial DC algorithm, the AC identification is carried out by establishing a number of different reference patterns, one associated to each subsystem. The failed subsystem is then identified as the one for which the reference pattern best matches the current pattern. Using PUA allows identifying the failed subsystem without previous training.

5.2.1. The Artificial DC Inputs

The inputs to the DC algorithm are the outcomes of the self/non-self discrimination over a moving time window of size T. Let the discrimination outcomes be defined as $Q_{\tau l}$ and $\overline{Q}_{\tau l}$, where $\tau = t-T, t-T+1, \ldots, t$ and $l = 1, 2, \ldots, N_p$, and let the current sample t discrimination matrix D_t and the complementary discrimination matrix $\overline{D}_t$ be defined by:

$$D_t = \begin{matrix} \\ t-T \\ t-T+1 \\ \\ t \end{matrix} \begin{matrix} \begin{matrix} S_1 & S_2 & \ldots & S_{N_p} \end{matrix} \\ \begin{bmatrix} Q_{11} & Q_{12} & \cdots & Q_{1N_p} \\ Q_{21} & Q_{22} & & Q_{2N_p} \\ \vdots & & \ddots & \vdots \\ Q_{t1} & Q_{t2} & \cdots & Q_{tN_p} \end{bmatrix} \end{matrix} \tag{5-12}$$

$$\overline{D}_t = \begin{matrix} \\ t-T \\ t-T+1 \\ \\ t \end{matrix} \begin{matrix} \begin{matrix} S_1 & S_2 & \ldots & S_{N_p} \end{matrix} \\ \begin{bmatrix} \overline{Q}_{11} & \overline{Q}_{12} & \cdots & \overline{Q}_{1N_p} \\ \overline{Q}_{21} & \overline{Q}_{22} & & \overline{Q}_{2N_p} \\ \vdots & & \ddots & \vdots \\ \overline{Q}_{t1} & \overline{Q}_{t2} & \cdots & \overline{Q}_{tN_p} \end{bmatrix} \end{matrix} \tag{5-13}$$

where $Q_{\tau l}$ and $\overline{Q}_{\tau l}$ are calculated using Eq. (5-10) and Eq. (5-11) respectively.

To address the AC evaluation problem, a new partition tracking matrix PT_t is introduced here. At each time sample t, the movement of the features over neighboring partition clusters is recorded over previous time window such that:

$$PT_t = \begin{matrix} \varphi_1 & \varphi_2 & \dots & \varphi_N \end{matrix} \begin{matrix} \\ \\ \\ \\ \end{matrix} \begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1N} \\ P_{21} & P_{22} & & P_{2N} \\ \vdots & & \ddots & \vdots \\ P_{t1} & P_{t2} & \cdots & P_{tN} \end{bmatrix} \begin{matrix} t-T \\ t-T+1 \\ \vdots \\ t \end{matrix} \tag{5-14}$$

where the row corresponding to feature i at time step t is defined as:

$$P_{ti} = P_{(t-1)i} + \Delta p_i \tag{5-15}$$

where Δp_i represents the change of feature i partition location. In other words, Δp_i can be calculated as:

$$\Delta p_i = \frac{\overline{\varphi}_{it} - \overline{\varphi}_{i(t-1)}}{\pi e_i} \tag{5-16}$$

where $\overline{\varphi}_{it}$ is the normalized value of the current feature i, $\overline{\varphi}_{i(t-1)}$ is the normalized value of feature i at previous step, and πe_i is the partition resolution of feature i for evaluation. Note that πe_i does not necessarily have to, but could, be equal to π_i.

5.2.2. The Artificial DC Components

The artificial DC [95] is a computational unit represented as a vector with eight components (Figure 5-1), as described next.

1. **The selection flag (λ_s)** represents the DC selection status, selected vs. not selected. A random number ρ is generated for each DC at each time step. Based on a prescribed selection rate σ between 0 and 1, the selection status of each DC is updated as:

$$\lambda_s = \begin{cases} 1 & if\ \rho \leq \sigma \\ 0 & if\ \rho > \sigma \end{cases} \tag{5-17}$$

where $\lambda_s = 1$ means that the DC is selected to process the self/non-self discrimination of the current time step.

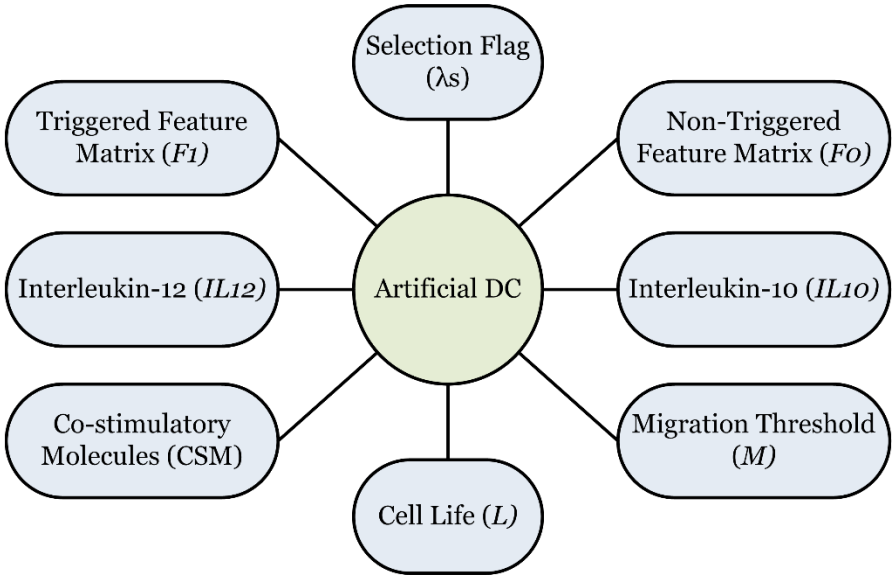

Figure 5-1. The Artificial DC Data Structure

2. **Co-stimulatory Molecules (CSM)** are proteins on the biological DC surface whose concentration increases whenever the DC processes a suspected antigen, regardless of the outcome. The corresponding first element of the artificial DC is a counter for the number of times a DC is selected from the pool and activated. The parameter is initialized as 0 and then updated as:

$$CSM_t = \begin{cases} CSM_{t-1} + 1 & \lambda_s = 1 \\ CSM_{t-1} & \lambda_s = 0 \end{cases} \tag{5-18}$$

3. **Interleukin-10 (IL10)** is a special compound produced by the biological DC when the tested entity is suspected to be part of the self. This parameter is initialized as 0 and updated whenever the DC is activated based on the complementary discrimination matrix:

$$IL10_t = \begin{cases} IL10_{t-1} + \Gamma_{10}(W_s^T, \overline{D}_t, W_0) & \lambda_s = 1 \\ IL10_{t-1} & \lambda_s = 0 \end{cases} \tag{5-19}$$

where:

$$W_0 = [w_{01} \quad w_{02} \quad \cdots \quad w_{0N_p}]^T \tag{5-20}$$

and:

$$W_s = [w_{s1} \quad w_{s2} \quad \cdots \quad w_{sT}]^T \tag{5-21}$$

W_0 and W_s are weighting factors; they allow considering potentially different sensitivity of the subsystems in capturing the fingerprint of the normal condition (W_0) and to assign different levels of priority to more recent data as opposed to older data (W_t). Γ_{10} is the interleukin-10 accumulation function.

For systems for which the AIS self is built as a set of homogenous dimensionality projections, with reduced hidden non-self regions, and with balanced distribution of projection discrimination capability [91], the interleukin-10 accumulation functions was defined as:

$$\Gamma_{10}\left(W_s^T, \overline{D}_t, W_0\right) = W_s^T \overline{D}_t W_0 \tag{5-22}$$

For the power system in this research effort, where the AIS self is built as a set of diverse dimensionality projections, the interleukin-10 accumulation functions was defined as:

$$\Gamma_{10}\left(W_s^T, \overline{D}_t, W_0\right) = \left\lfloor \frac{\sum(W_s^T \overline{D}_t W_0)}{N_p} \right\rfloor \tag{5-23}$$

4. **Interleukin-12 (IL12)** is a special compound produced by the biological DC when the tested entity is suspected to be an antigen. This parameter is initialized as 0 and updated whenever the DC is activated based on the discrimination matrix:

$$IL12_t = \begin{cases} IL12_{t-1} + \Gamma_{12}(W_s^T, D_t, W_1) & \lambda_s = 1 \\ IL12_{t-1} & \lambda_s = 0 \end{cases} \tag{5-24}$$

where:

$$W_1 = [w_{11} \quad w_{12} \quad \cdots \quad w_{1N_p}]^T \tag{5-25}$$

W_1 is a weighting vector that puts different levels of confidence for each subsystem regarding its capability in capturing the fingerprint of the abnormal condition. The Γ_{12} is the interleukin-12 accumulation function.

For systems for which the AIS self is built as a set of uniform dimensionality projections, with reduced hidden non-self regions, and with balanced distribution of projection discrimination capability [91], the interleukin-12 accumulation functions can be defined as:

$$\Gamma_{12}(W_s^T, D_t, W_1) = W_s^T D_t W_1 \quad (5\text{-}26)$$

For the power system in this research effort, where the AIS self is built as a set of diverse dimensionality projections, the interleukin-12 accumulation functions was defined as:

$$\Gamma_{12}(W_s^T, D_t, W_1) = \left\lceil \frac{\sum(W_s^T D_t W_1)}{N_p} \right\rceil \quad (5\text{-}27)$$

5. **Cell life** ($\mathcal{L}$) is a parameter reflecting the fact that cells experience a healthy programmed cell death. It is initialized as a random integer representing the total number of activations that the cell can support before re-initialization/replacement. In other words, an artificial DC is considered dead when its CSM exceeds $\mathcal{L}$.Each DC updates its life property according to:

$$\mathcal{L}_t = \mathcal{L}_{t-1} - 1 \quad (5\text{-}28)$$

The cell life parameter allows only recently generated/processed cells to stay in the pool and, therefore; it eliminates old information.

6. **Migration threshold ($\mathcal{M}$)** represents the duration of the process of acquiring antigen information before transferring it for further processing. For the biological cells, maturity is reached when the CSM achieves a certain level. In the artificial DC, this parameter is initialized as a random integer. The migration threshold is initialized to a random integer and updated as following:

$$\mathcal{M}_t = \begin{cases} \mathcal{M}_{t-1} - 1 & \Omega = 1 \\ \mathcal{M}_{t-1} & \Omega = 0 \end{cases} \quad (5\text{-}29)$$

where $\Omega = 1$ when the current sample indicates change in operational condition. In other words, $\Omega = 1$ if the current detection status is normal, while the current sample indicates abnormal condition, or the current detection status is abnormal and the current sample indicates normal conditions. If the current detection status and the current sample indicate the same status, then $\Omega = 0$. The update of migration threshold allows a shorter detection time and lower false alarms when the system transits between normal and abnormal conditions and vice versa. An artificial DC is said to be mature when its CSM reaches the migration threshold.

7. **Triggered and Non-triggered Features Matrices.** A biological DC is capable of engulfing an intruding entity and breaking it up into constituent components. Upon maturity, this information is transferred to the adaptive immune system and used for counter-action. Similarly, an artificial DC presents information about the processed input by constructing the triggered features matrix $F1$. Unlike the previous version of the DC algorithm, the $F1$ matrix consist of two layers. The first layer of the triggered feature matrix (F_1^1) is used to record the number of times subsystem S_l is triggered and is defined as:

$$F_1^1 = \begin{matrix} S_1 & S_2 & \dots & S_{N_p} \end{matrix} \\ \begin{bmatrix} \psi_{11}^1 & \psi_{12}^1 & \cdots & \psi_{1N_p}^1 \\ \psi_{21}^1 & \psi_{22}^1 & & \psi_{2N_p}^1 \\ \vdots & & \ddots & \vdots \\ \psi_{N1}^1 & \psi_{N2}^1 & \cdots & \psi_{NN_p}^1 \end{bmatrix} \begin{matrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_N \end{matrix} \tag{5-30}$$

where the matrix elements ψ_{il}^1 are initialized to zeros and updated if $\varphi_i$is one of the feature coordinates of the triggered subsystem S_l. In other words:

$$\psi_{il}^1 = \begin{cases} \psi_{il}^1 + 1 & if\ S_l\ is\ triggred\ and\ \alpha_{il} = 1 \\ \psi_{il}^1 & otherwise \end{cases} \tag{5-31}$$

The second layer of the triggered feature matrix (F_1^2) captures the number of partitions over which each feature has moved. This second layer is defined as:

$$F_1^2 = \begin{matrix} S_1 & S_2 & \dots & S_{N_p} \end{matrix} \\ \begin{bmatrix} \psi_{11}^2 & \psi_{12}^2 & \cdots & \psi_{1N_p}^2 \\ \psi_{21}^2 & \psi_{22}^2 & & \psi_{2N_p}^2 \\ \vdots & & \ddots & \vdots \\ \psi_{N1}^2 & \psi_{N2}^2 & \cdots & \psi_{NN_p}^2 \end{bmatrix} \begin{matrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_N \end{matrix} \tag{5-32}$$

where the matrix elements ψ_{il}^2 are initialized to zeros and updated with the partition tracking matrix elements if φ_i is one of the feature coordinates of the triggered subsystem S_l. In other words:

$$\psi_{il}^2 = \begin{cases} \psi_{il}^2 + P_{ti} & if\ S_l\ is\ triggred\ and\ \alpha_{il} = 1 \\ \psi_{il}^2 & otherwise \end{cases} \tag{5-33}$$

The non-triggered features matrix (F_0) is recording the number of times subsystem S_l is not triggered. The matrix is defined as:

$$F_0 = \begin{matrix} \\ \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_N \end{matrix} \begin{matrix} \begin{matrix} S_1 & S_2 & \dots & S_{N_p} \end{matrix} \\ \begin{bmatrix} \psi_{11}^0 & \psi_{12}^0 & \cdots & \psi_{1N_p}^0 \\ \psi_{21}^0 & \psi_{22}^0 & & \psi_{2N_p}^0 \\ \vdots & & \ddots & \vdots \\ \psi_{N1}^0 & \psi_{N2}^0 & \cdots & \psi_{NN_p}^0 \end{bmatrix} \end{matrix} \tag{5-34}$$

where the matrix elements are initialized to zeros and updated if $\varphi_i$is one of the feature coordinates of the non-triggered subsystem S_l. In other words:

$$\psi_{il}^0 = \begin{cases} \psi_{il}^0 + 1 & if\ S_l\ is\ nontriggred\ and\ \alpha_{il} = 1 \\ \psi_{il}^0 & otherwise \end{cases} \tag{5-35}$$

5.2.3. The Artificial DC Algorithm

The algorithm starts by initializing a set $\mathbb{C}$ of a pre-defined number N_{DC} of immature DCs with default properties. It is worth mentioning that the algorithm starts after the first time window of size T has passed. At each time step, the current measured features point undergoes the self/non-self discrimination process to update the discrimination and partition tracking matrices (D_t,$\overline{D}_t$ and PT_t). At the same time, a random number of DCs is selected from the pool and each selected DC processes the discrimination and partition tracking matrices and updates its components. The term *mature DCs* indicates that the selected DCs co-stimulatory molecules (CSM) parameter has reached the migration threshold $\mathcal{M}$. The collection of mature DCs is processed through the ACDIE logic to determine the system status. The term *dead DCs* refers to all DCs with a life parameter $\mathcal{L}$ equal to zero. Any mature or dead DCs are replaced in the set $\mathbb{C}$ by new DCs with default properties. The flow chart of the artificial DC algorithm for ACDIE is presented in Figure 5-2.

5.3. Abnormal Conditions Detection

Detecting the presence of an AC is a critical step in the ACM process. A reliable and fast detection scheme is required for the AC identification, evaluation, and accommodation schemes to provide timely and accurate outcomes. The AC detection logic starts by processing the collection of mature DCs. Any migrated DC with $IL12 \geq IL10$ is called stimulatory DC, since it activates the production of cytotoxic T-cells.

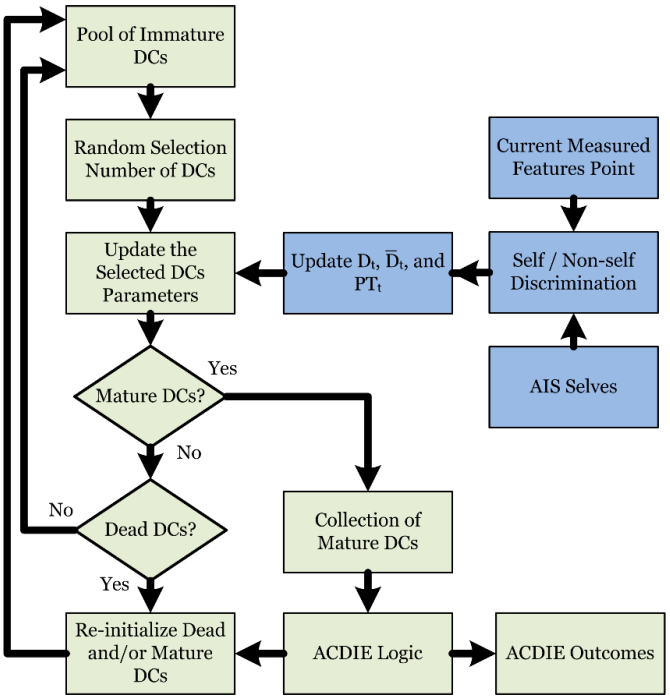

Figure 5-2. The Artificial DC Algorithm

The set of activated stimulatory T-cells K can be expressed as:

$$K = \{K_i \in \mathbb{N} \quad | \quad i = 1,2, \dots, N\} \tag{5-36}$$

$$K_i = \sum_{n=1}^{N_{sdc}} \sum_{l=1}^{N_s} \psi_{il}^1 \tag{5-37}$$

where $\mathbb{N}$ is the set of natural numbers, N_{sdc} is the number of stimulatory DCs and K_i is the number of stimulatory T-cells corresponding to feature φ_i.

Any migrated DC with $IL12 < IL10$ is called regulatory DC since it activates the production of suppressor T-cells. The set of activated suppressor T-cells R can be expressed as:

$$R = \{R_i \in \mathbb{N} \quad | \quad i = 1,2, \dots, N\} \tag{5-38}$$

$$R_i = \sum_{m=1}^{N_{rdc}} \sum_{l=1}^{N_s} \psi_{il}^0 \tag{5-39}$$

where N_{rdc} is the number of regulatory DCs and R_i is the number of suppressor T-cells corresponding to feature φ_i.

The role of the suppressor T-cells is to regulate the adaptive immune response by suppressing an equal number of activated cytotoxic T-cells, which results in a set of residual cytotoxic T-cells given by:

$$\widetilde{K} = \{\widetilde{K}_i = K_i - R_i \quad | \quad i = 1,2,\dots,N\} \tag{5-40}$$

This stimulation/suppression of the adaptive immune system as determined by the production of cytotoxic and suppressor T-cells indicates whether the system is under normal or abnormal conditions. Therefore, the overall detection outcome at any sample time t can be calculated using the following equation:

$$Det_t = \begin{cases} 0 & \sum_{i=1}^{N} \widetilde{K}_i \leq 0 \\ 1 & otherwise \end{cases} \tag{5-41}$$

The DC algorithm for AC detection is illustrated in Figure 5-3.

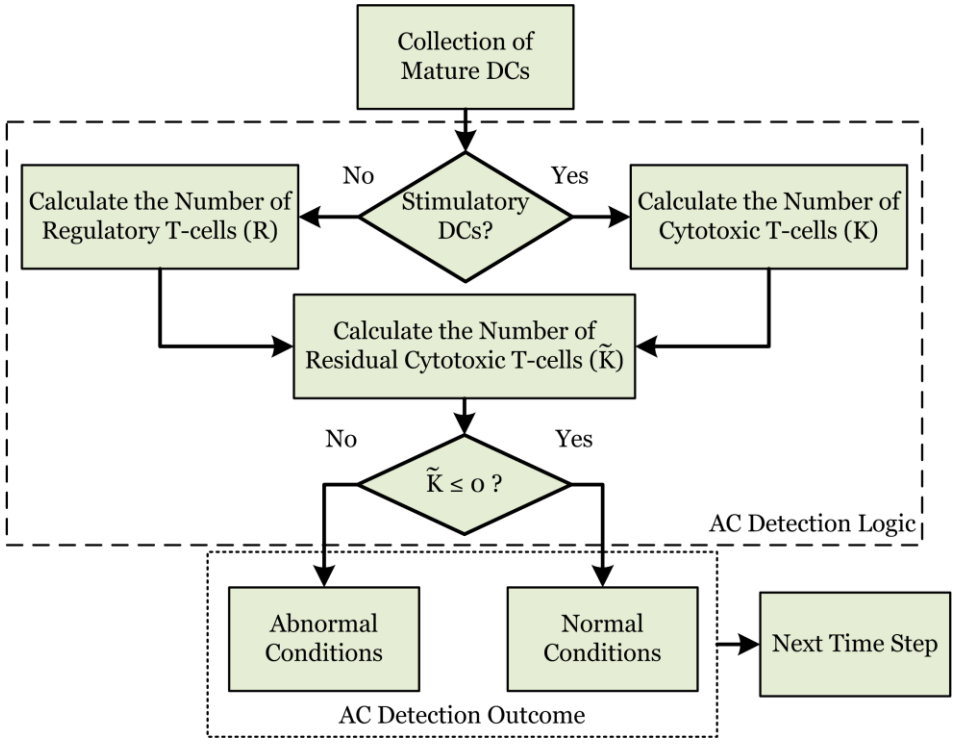

Figure 5-3. The Artificial DC Algorithm for AC Detection

If the detection outcome is zero, the algorithm proceeds to the next time sample; otherwise, the algorithm determines the identification and evaluation outcome before proceeding to the next time step.

5.4. The Naïve Bayes Classifier

Some of the approaches described below require a pattern recognition algorithm to determine the best match to the current patterns produced by the migrated DCs. The naïve Bayes classifier [96] is one of the most popular pattern recognition algorithms. Naïve Bayes classifier is a probabilistic supervised learning algorithm which provides very high classification rate with very fast training and validation phases. These advantages of the classifier make it the most suitable algorithm for the purpose of pattern recognition for the AC identification and evaluation problem.

Let $R = [r_s \mid s = 1,2, \dots\ a]^T$ be an $a \times 1$ vector of continuous values of a attributes to be classified into a class variable $M = \{m_c \mid\ c = 1,2, \dots, b\}$.

For each class m_c the sample mean vector μ_c and the sample covariance matrix Σ_c can be calculated from a given sample attribute vectors. Assume the vectors $R_c^{(n)} = \left[{r_{sc}}^{(n)} \mid s = 1,2, \dots\ a\right]$ is the n^{th} sample vector and it belongs to class m_c, then:

$$\mu_c = \frac{1}{nt_c} \sum_{n=1}^{nt_c} {r_{sc}}^{(n)} \tag{5-42}$$

$$\Sigma_c = \frac{1}{nt_c - 1} \sum_{n=1}^{nt_c} ({r_{sc}}^{(n)} - \mu_c)({r_{sc}}^{(n)} - \mu_c)^T \tag{5-43}$$

where nt_c the number of samples in class m_c.

Once the mean vectors and covariance matrices for all classes are calculated, the next equation can be used to calculate the quadratic discriminant function Δ_c such that:

$$\Delta_c(\tilde{R}) = ln(nt_c) - \frac{1}{2} ln|\Sigma_c| - \frac{1}{2}\left(\tilde{R} - \mu_c\right)^T \Sigma_c^{-1} \left(\tilde{R} - \mu_c\right) \tag{5-44}$$

where $\tilde{R} = [\tilde{r}_s \mid s = 1,2, \dots\ a]^T$ is the vector of continuous values of a attributes to be classified into b classes

Once the quadratic discriminant function Δ_c is calculated for all the b classes, the class $\tilde{c}$ of the current attributes vector $\tilde{R}$ can then be identified using:

$$\tilde{c} = \underset{c=1,2,\dots,b}{argmax}(\Delta_c(\tilde{R})) \tag{5-45}$$

5.5. Abnormal Conditions Identification

The AC identification is the process that isolates the subsystem that is mostly affected by or is the source of the detected AC. The identification process is a critical step in ACM for an accurate and reliable evaluation and accommodation. The migrated DCs in the artificial DC mechanism presented in section 5.2.3 carry information that is useful in identifying the failed subsystem. This information is summarized by the triggered-features matrix (F_1) defined in Eq.(5-30). Different patterns can be extracted from the F_1 matrices of the migrated DCs, depending on how the matrices are viewed. The next sections present two different approaches to address the AC identification problem.

5.5.1. The Subsystem Pattern Approach

In this approach [97], features that represent one subsystem are grouped together to form the self of that subsystem. In other word, each self projection represents an actual subsystem such that $N_p = N_s$, where N_p is number of self projections and N_s is the number of targeted subsystems. Features could belong to one or more subsystems at the same time; however, the approach best applies to systems that exhibit minimum features overlapping as possible. Ideally, the sum of self projections-features mapping matrix SF rows (Eq. (5-7)) is equal to a column unit vector.

Let N be the total number of selected features, and N_s be the total number of subsystems. The size of subsystem k, z_k represents the number of features in that subsystem or the sum of the k^{th} column in subsystem feature mapping matrix SF (Eq. (5-7)). The construction of the subsystem pattern starts by defining the elements of the subsystem pattern such that:

$$SP = \{sp_k \mid k = 1,2,\ N_p\} \tag{5-46}$$

where SP elements are calculated by summing the elements of the first layer of the triggered matrix F_1^1 rows, such that:

$$SP = \{\sum_{a=1}^{N_{mdc}} \sum_{i=1}^{N} \psi_{ik}^1 \mid k = 1,2,\ N_p\} \tag{5-47}$$

where N_{mdc} is the number of migrated DCs. The normalized subsystem pattern $\overline{SP}$ is calculated by dividing the subsystem pattern element by the corresponding subsystem size such that:

$$\overline{SP} = \left\{ {}^{sp_k}/_{Z_k} \,\middle|\, l = 1,2,\ N_p \right\} \tag{5-48}$$

The failed subsystem index k at each time sample t $(\widetilde{k_t})$ is then identified by locating the index of maximum normalized subsystem pattern $\overline{SP}$ such that:

$$\widetilde{k_t} = \underset{k=1,2,\ldots,N_s}{argmax}(\overline{SP}) \tag{5-49}$$

If more than one k satisfies the above equation, previous time sample results may be used to determine $\widetilde{k_t}$. Once $\widetilde{k_t}$ is calculated, the outcome of the AC identification Idn_t is obtained as:

$$Idn_t = \{id_k |\, k = 1,2,\ N_s\} \tag{5-50}$$

where:

$$id_k = \begin{cases} 1 & k = \widetilde{k_t} \\ 0 & otherwise \end{cases} \tag{5-51}$$

The subsystem pattern approach does not require training or the use of pattern matching algorithm, if the features are carefully organized into the different subsystems and a minimal feature overlapping was ensured. If the subsystem pattern is not applicable, then the feature pattern approach could be used to address the AC identification problem.

5.5.2. The Features Pattern Approach

In this approach [91] N_s different reference patterns, one associated to each subsystem, must be generated offline using training test data. Current patterns are generated online either during a simulation test or from recorded validation tests. Therefore, AC identification process can be seen as a pattern recognition problem in which the failed subsystem is identified as the one for which the reference pattern best matches the current pattern.

At each sample time, after an AC is detected, the features pattern can be obtained from the triggered-feature matrices of all migrated DCs. The construction of the feature pattern starts defining the elements of the subsystem pattern such that:

$$FP = \{fp_i \,|\, i = 1,2,\ N\} \tag{5-52}$$

where FP element are calculated by summing the elements of the first layer of the triggered matrix F_1^1 columns such that:

$$FP = \left\{\sum_{a=1}^{N_{mdc}} \sum_{k=1}^{N_s} \psi_{ik}^1 \,\middle|\, i = 1,2,\ N\right\} \tag{5-53}$$

where N_{mdc} is the number of migrated DCs. The normalized feature pattern $\overline{FP}$ is calculated by dividing the features pattern element by the corresponding subsystem size such that:

$$\overline{FP} = \frac{FP}{||FP||} \tag{5-54}$$

where the norm in the denominator is the Euclidean norm. At this point, the naïve Bayes classifier algorithm is required in order to determine which of the N_s reference patterns.is the closest to the current pattern and ultimately constitute the outcome of the AC identification process as:

$$Idn_t = \{id_k |\ k = 1,2,\ N_s\} \tag{5-55}$$

where:

$$id_k = \begin{cases} 1 & \overline{FP}\ best\ matches\ k^{th}\ pattern \\ 0 & otherwise \end{cases} \tag{5-56}$$

The reference patterns, in this case the mean vectors and covariance matrices, are obtained by training the classifier offline against samples from a set of training tests as presented in Figure 5-4. The training phase is achieved by completing the following steps:

1. For each subsystem k, prepare a set of training tests where the subsystem is affected by one AC.

2. For each training test, run the DC algorithm for detection until the AC is detected, then calculate $R = \overline{FP}$ using Eq. (5-54)

3. Denote the number of R vectors for all training tests by nt_k , use $c = k$, and compute the mean vector μ_k, Σ_k and sample covariance matrix using Eq. (5-42) and Eq.(5-43) respectively.

4. Save the nt_k, μ_k, Σ_k as to the identification library of reference patterns

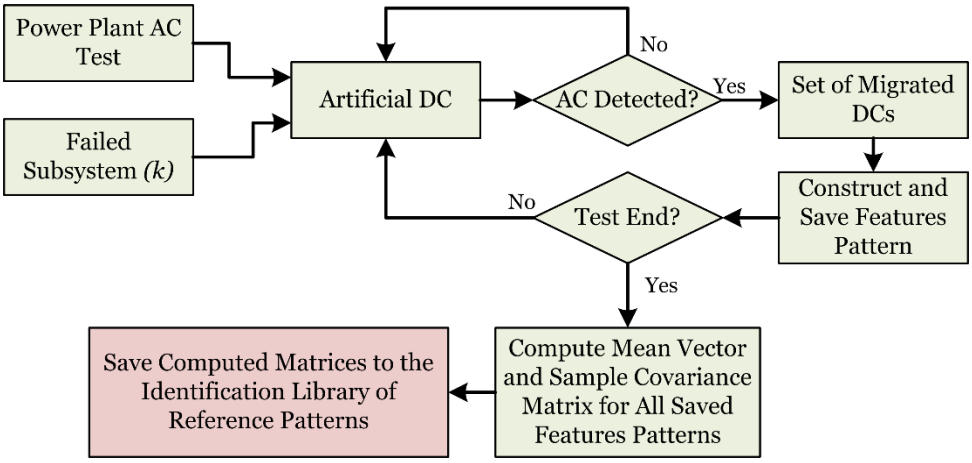

Figure 5-4. Training the Naïve Bayes for AC Identification

After calculating all the N_s reference patterns, the Naïve Bayes classifier can be used online to identify the failed subsystem from using current feature patterns in a given AC test as presented in Figure 5-5.

1. Once an AC has been detected, use Eq. (5-54) to calculated the current feature pattern $\tilde{R} = \overline{FP}$

2. For each subsystem k, calculate Δ_k using Eq. (5-44) using the mean vectors μ_k, and sample covariance matrices Σ_k calculated in the training phase.

3. Determine the failed subsystem using Eq. (5-45).

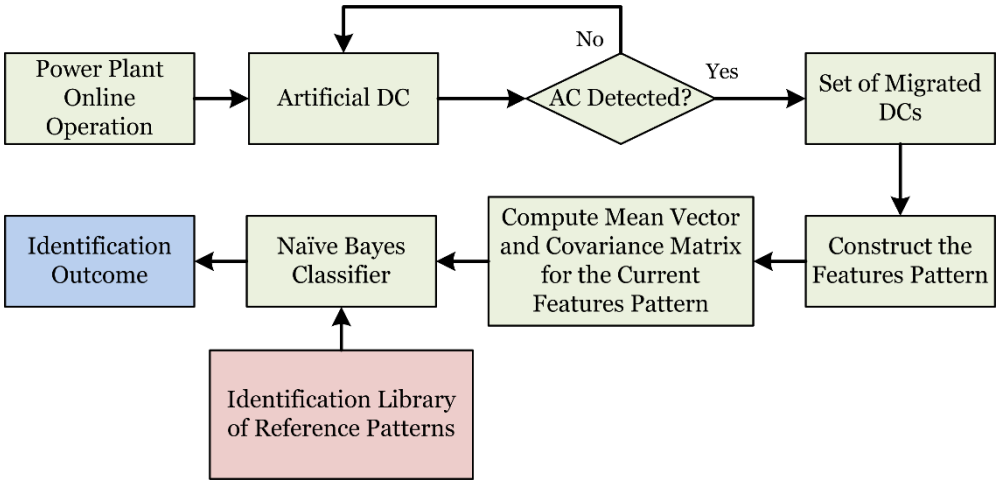

Figure 5-5. Online AC Identification Using Features Pattern Approach

5.6. Abnormal Conditions Evaluation

AC evaluation consists of determining the type of failure (i.e. qualitative evaluation), estimating the severity of the failure (i.e. direct quantitative evaluation), and

assessing the effects of the AC on the plant operation constraints (i.e. indirect quantitative evaluation). In general, the outcome of the evaluation process EO can be formulated as:

$$EO = \{EvQ\ EvDQ\ EvIQ\} \tag{5-57}$$

In this research, only direct evaluation was performed relative to AC type and severity based on categorical metrics. In the past, the approach using the DC mechanism for identification using features pattern approach presented in 5.5.2 was extended [98] to address the AC qualitative and direct quantitative evaluation. A different approach for building the pattern is introduced here to address the evaluation problem while taking advantages of the high dimensional self built with the PUA.

5.6.1. The Partition Tracking Pattern Approach

The first AC evaluation logic using this approach assumes accurate identification outcome and uses only the features associated with the failed subsystem to determine the AC type and severity. Using this approach minimizes the computational issues associated with generating and saving the evaluation reference patterns. However, since the approach uses the features associated with the failed subsystem, incorrect identification outcome will mean incorrect evaluation results. Alternative approach is presented at the end of this section to overcome this problem.

Let the number of the targeted subsystem be N_s such that $k = 1,2, \dots, N_s$, and let the number of AC types affecting the k^{th} subsystem be ACt_k such that $n_k = 1,2, \dots, ACt_k$, and let the severity scale of the n^{th} AC type of the k^{th} subsystem be ACs_{kn} such that $s_{kn} = 1,2, \dots, ACs_{kn}$. The total number of reference evaluation patterns EP required to address all the AC types and severities are:

$$EP = \sum_{k=1}^{N_s} \sum_{n=1}^{ACt_k} n * ACs_{kn} \tag{5-58}$$

Once the AC detection and identification outcomes have been determined, the partition tracking pattern (PTP) is constructed using the second layer of the triggered feature matrix F_1^2 and the features of the failed subsystem k such that:

$$PTP = \left\{ \sum_{a=1}^{N_{mdc}} \psi_{ik}^2 \,\middle|\, i = 1,2,\ N\ and\ \alpha_{ik} = 1 \right\} \tag{5-59}$$

where N_{mdc} is the number of migrated DCs. The normalized partition tracking pattern $\overline{PTP}$ is calculated using:

$$\overline{PTP} = \frac{PTP}{N_{mdc}} \tag{5-60}$$

At this point, the naïve Bayes classifier algorithm is required in order to determine the class in which the current pattern best matches one of the EP evaluation reference patterns. Ultimately, the outcome of the AC evaluation is:

$$EvQ = \{the\ AC\ type\ of\ best\ pattern\ match\} \tag{5-61}$$

and:

$$EvDQ = \{the\ AC\ severity\ of\ best\ pattern\ match\} \tag{5-62}$$

The evaluation reference patterns are obtained by training the classifier offline against samples from a set of training tests as presented in Figure 5-6.

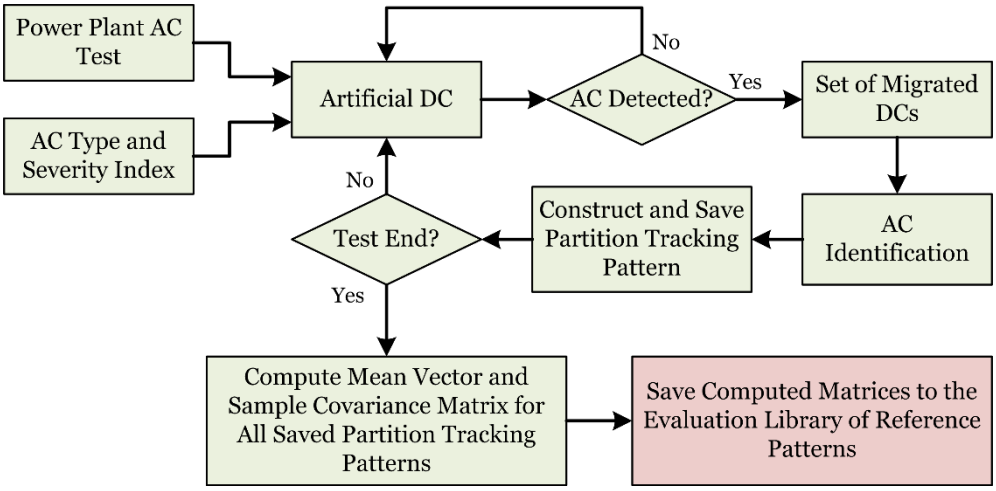

Figure 5-6. Training the Naïve Bayes for AC Evaluation

The training phase is achieved by completing the following steps:

1. For each subsystem k, prepare a set of $\sum_{n=1}^{ACt_k} n * ACs_{kn}$ set training tests. In each test, subsystem k is affected by one AC type and severity. Identify the AC type and severity combinations by a unique index d.

2. For each training test, run the DC algorithm for detection until the AC is detected and identified then calculate $R = \overline{PTP}$ using Eq. (5-60).

3. Denote the number of R vectors for all training tests by nt_d , use $c = d$, and compute the mean vector μ_d and covariance matrix Σ_d using Eq. (5-42) and Eq.(5-43), respectively.

4. Save the computed nt_d, μ_d, Σ_d to the evaluation library of reference patterns.

After saving all the EP reference patterns, the naïve Bayes classifier can be used online to identify the AC type and severity using the partition tracking patterns in a given AC test, as presented in Figure 5-7.

1. Once an AC have been detected and the affected subsystem identified, use Eq. (5-60)to calculated the current feature pattern $\tilde{R} = \overline{PTP}$

2. For each type and severity index d, calculate Δ_d using Eq. (5-44) with the mean vectors μ_d, and sample covariance matrices Σ_d calculated in the training phase.

3. Determine the AC type and severity class using Eq. (5-45).

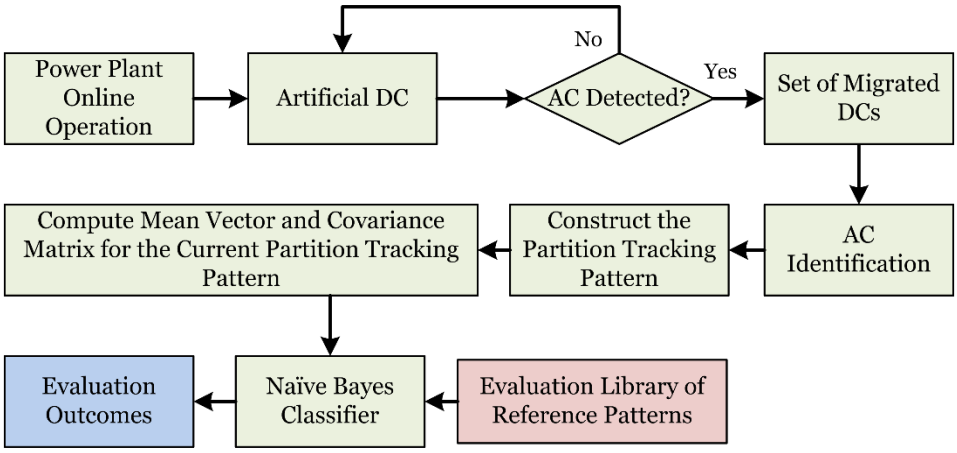

Figure 5-7. Online AC Evaluation Using Partition Tracking Pattern Approach

5.6.2. The Feature Pattern Approach

A similar approach to the one used for the AC identification problem can be applied for AC qualitative and direct evaluation purposes. EP patterns, each corresponding to an AC type and severity of the affected subsystem, are established using Eq. (5-54). The patterns are to be defined based on the detection outcomes of all self projections, which are summarized by the first layer of the triggered matrix F_1^1 matrix provided by the migrated DCs. Readers are referred to references [91], [99] for more information

regarding this approach and other approaches to address the AC identification and evaluation problem using AIS.

5.7. One Step AC Identification and Evaluation

The one step AC identification and evaluation approach establishes EP patterns using all available features or projections. At each time step, the current pattern is compared against all the EP patterns and the identification and evaluation outcomes are extracted from the best match. In this case, the PTP becomes:

$$PTP = \left\{ \sum_{a=1}^{N_{mdc}} \sum_{k=1}^{N} \psi_{ik}^{2} \,\middle|\, i = 1,2,\ N \right\} \tag{5-63}$$

All other aspects of the identification and evaluation processes will be the same. However, closer attention should be paid to the computational issues associated with pattern construction, saving, and matching, especially with a system with a high number of features.

The training phase is achieved by completing the following steps, as presented in Figure 5-8:

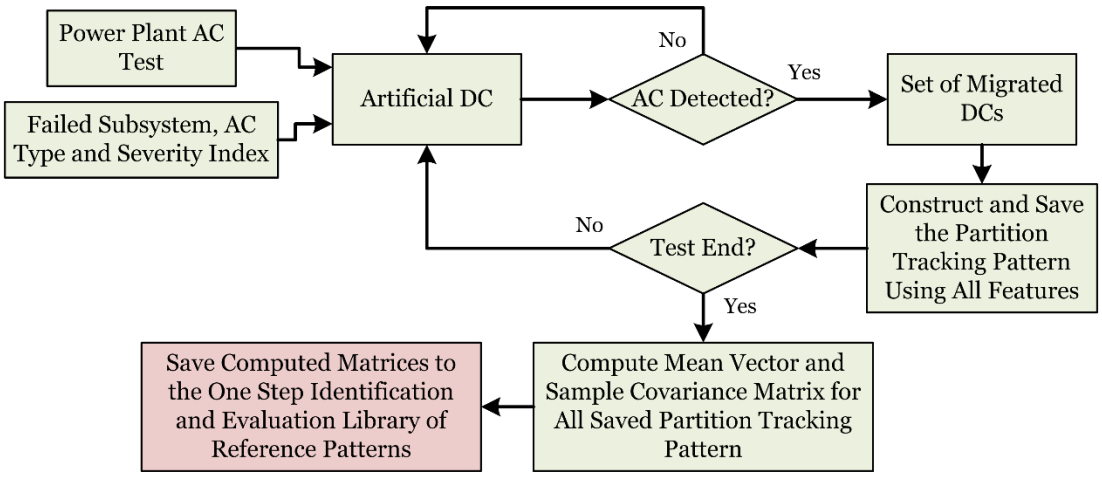

Figure 5-8. Training the Naïve Bayes for One Step AC Identification and Evaluation

1. For each subsystem k, prepare a set of $\sum_{n=1}^{ACt_k} n * ACs_{kn}$ set training tests. In each test, subsystem k is affected by one AC type and severity. Identify the failed subsystem k, AC type, and severity combinations by a unique index e.

2. For each training test, run the DC algorithm for detection until the AC is detected then calculate $R = \overline{PTP}$ using Eq. (5-63).

3. Denote the number of R vectors for all training tests by nt_e , use $c = e$, and compute the mean vector μ_e and covariance matrix Σ_e using Eq. (5-42) and Eq.(5-43) respectively.

4. Save the computed nt_e, μ_e, Σ_e to the evaluation library of one step identification and evaluation reference patterns.

After saving all the EP reference patterns, the naïve Bayes classifier can be used online to identify the failed subsystem, the AC type and severity using the partition tracking patterns in a given AC test using the following steps as presented in Figure 5-9:

1. Once an AC has been detected, use Eq. (5-64) to calculate the current feature pattern $\tilde{R} = \overline{PTP}$

2. For each subsystem, AC type and severity index e, calculate Δ_e using Eq. (5-44) with the mean vectors μ_e, and sample covariance matrices Σ_e calculated in the training phase.

3. Determine the unique subsystem, and AC type and severity index using Eq. (5-45).

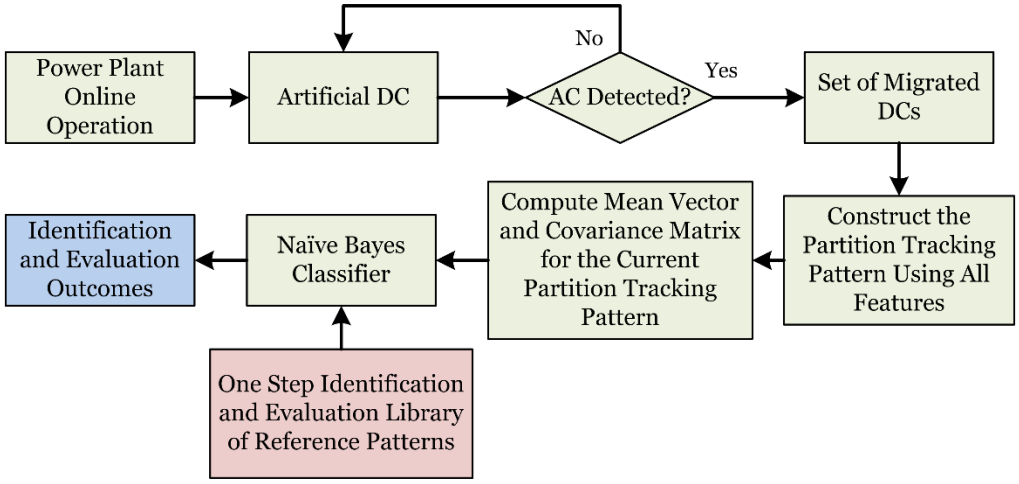

Figure 5-9. Online One Step AC Identification and Evaluation

5.8. ACDIE Performance Indices

The performance of the ACDIE scheme is evaluated using six indices: false alarm rate (FA), detection time (DT), detection rate (DR), identification rate (IR), type evaluation rate (TER), and severity evaluation rate (SER).

In a normal test, assume the number of samples in which detection outcome Det_t has a value equal to zero (true negatives) to be NS_T, and let the total number of samples in which detection outcome Det_t has a value equal to one (false positives) to be NS_F. The FA is then calculated using:

$$FA = \frac{NS_F}{NS_F + NS_T} \times 100 \tag{5-64}$$

the FA is obviously calculated only in tests under normal operations. Ideally, the FA should be zero. Non-zero FA might indicate incomplete self data, selection of small partitioning size, or selection of small time window size in the DC algorithm parameters.

DT is the time difference in seconds between the time of the AC occurrence and the time of its successful detection. Assuming TOC is the AC time of occurrence, and TOD is the AC time of detection, then DT is calculated using:

$$DT = TOD - TOC \tag{5-65}$$

Small DT indicates good detection performance. Larger DT might indicate selection of large partitioning size, or selection of large time window size in the DC algorithm parameters, or improper features selection.

DR is defined as the percentage ratio between the number of samples detected as abnormal and the total number of samples in a validation test under abnormal conditions calculated after first successful detection. In an abnormal test, assume the number of samples calculated after DT in which detection outcome Det_t has a value equal to one to (true positives) be FS_T, and let the total number of samples calculated after DT in which detection outcome Det_t has a value equal to zero (false negatives) to be FS_F. The DR is then calculated using:

$$DR = \frac{FS_T}{FS_T + FS_F} \times 100 \tag{5-66}$$

IR is the ratio between the number of samples in which the algorithm correctly identified the subsystem under AC and the total number of samples in the test after a successful detection. In an abnormal test, assume FS_{IT} is the number of samples

calculated after TOD in which detection outcome Det_t has a value equal to one and the identification algorithm provided correct outcomes. Let FS_{IF} to be the total number of samples calculated after TOD in which detection outcome Det_t has a value equal to one and the identification algorithm provided incorrect outcomes. The IR is then calculated using:

$$IR = \frac{FS_{IT}}{FS_{IT} + FS_{IF}} \times 100 \quad (5\text{-}67)$$

TER is the ratio between the number of samples in which the evaluation algorithm correctly evaluated the AC type and the total number of samples in the test calculated after TOD given correct detection and identification outcomes. In an abnormal test, assume FS_{TT} is the number of samples calculated after TOD in which detection outcome Det_t has a value equal to one, the identification algorithm provided correct outcomes, and the evaluation algorithm provided correct AC type estimation. Let FS_{TF} be the total number of samples calculated after TOD in which detection outcome Det_t has a value equal to one, the identification algorithm provided correct outcomes, and the evaluation algorithm provided incorrect AC type estimation The TER is then calculated using:

$$TER = \frac{FS_{TT}}{FS_{TT} + FS_{TF}} \times 100 \quad (5\text{-}68)$$

Finally, SER is the ratio between the number of samples in which the algorithm correctly evaluated the AC severity to the total number of samples in the test in which all preceding algorithms provided correct outcomes. In an abnormal test, assume FS_{ST} is the number of samples calculated after TOD in which detection outcome Det_t has a value equal to one, the identification algorithm provided correct outcomes, and the evaluation algorithm provided correct AC type and severity estimation. Let FS_{SF} be the total number of samples calculated after TOD in which detection outcome Det_t has a value equal to one, the identification algorithm provided correct outcomes, and the evaluation algorithm provided correct AC type, but incorrect severity estimation The SER is then calculated using:

$$SER = \frac{FS_{ST}}{FS_{ST} + FS_{SF}} \times 100 \quad (5\text{-}69)$$

Ideally, all of IR, TER, and SER should be 100%. Small rates might indicate the improper selection of subsystem features, or incomplete reference patterns.

For a high performance ACDIE algorithm, all DR, IR, TER, and SER values should be 100%. Smaller values might indicate improper selection of features, or selection of wrong partitioning size, or incorrect DC parameters settings.

Chapter 6. Abnormal Conditions Accommodation

This chapter introduces the use of the AIS paradigm in conjunction with other artificial intelligence techniques, such as neural networks and fuzzy logic, to develop adaptive control mechanisms that are expected to enhance the performance of a baseline controller for normal and abnormal operational conditions.

The operational strategy and the architecture of the biomimetic adaptive controller are envisioned to rely on close interaction with the immunity-based ACDIE. It is expected that the performance of the system will be enhanced by combining an analytical adaptive component based on the immune humoral feedback mechanism with the information stored in the artificial immune system, which is primarily used for abnormal condition detection and identification. Figure 6-1 provides a high level insight into the architecture and operation of the biomimetic adaptive control laws.

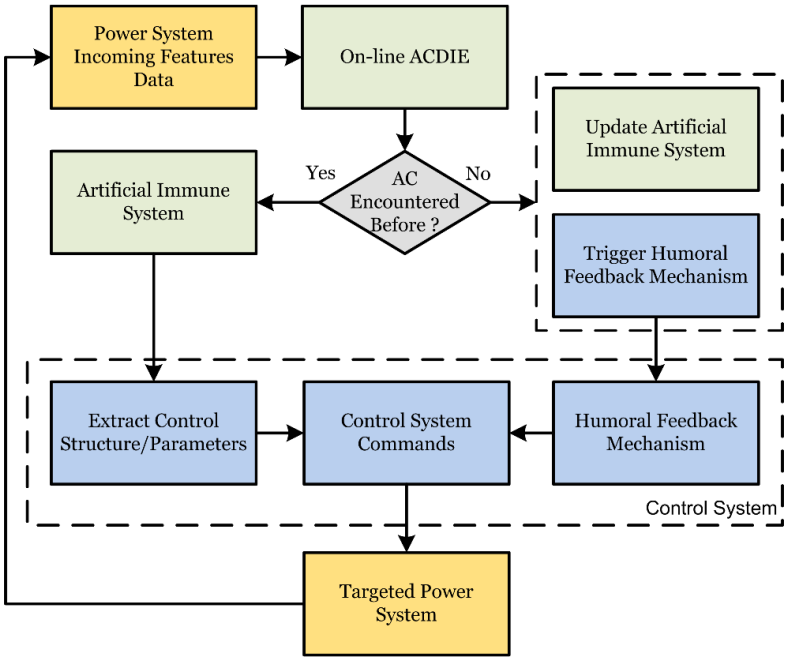

Figure 6-1. Architecture of the Biomimetic Adaptive Control Laws

The on-line ACDIE should provide timely and accurate outcomes for an effective accommodation process. Once an AC has been diagnosed successfully, the control system utilizes that information to determine the best accommodation strategy. If the AC has never been encountered before, the control system uses an adaptive control augmentation inspired from the humoral feedback mechanism and updates the artificial immune system framework with the accommodation parameters and the system response for

future optimization and uses. If the AC has been encountered before, then the control system uses a pre-defined and optimized control strategy and parameters to accommodate the AC. In this research, two main solutions for the adaptive biomimetic mechanism have been investigated: the artificial neural network (ANN) mechanism and the immunity humoral feedback mechanism.

6.1. The Artificial Neural Net Adaptive Controller

The artificial neural net (ANN)-based adaptive mechanism [100], [101] relies on the capability of the ANNs to model/approximate functions. In this case, the function approximated is the modification of the system due to the abnormal condition. A single-hidden-layer ANN with on-line training was considered. The architecture of the adaptive control laws using ANN to augment a baseline controller is presented in Figure 6-2.

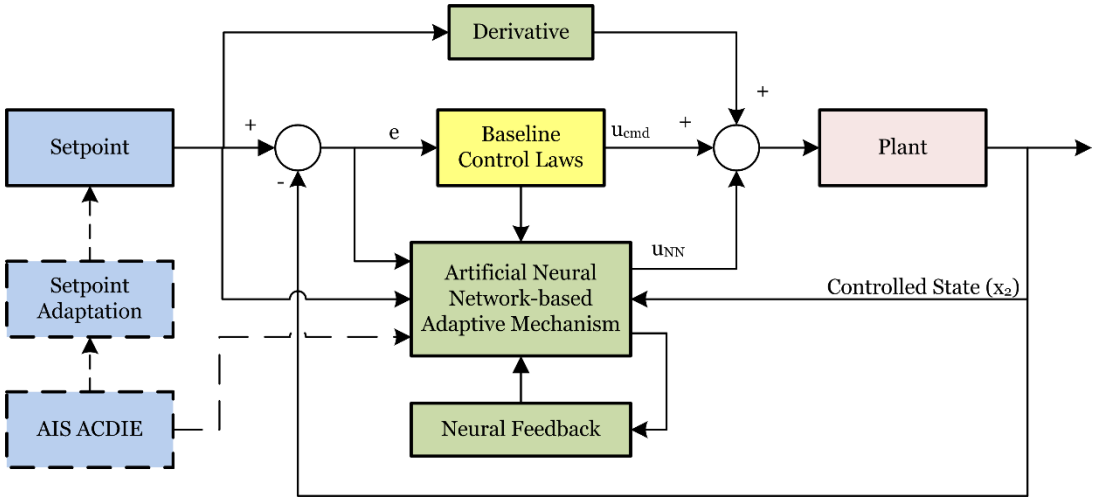

Figure 6-2. Adaptive Control Based on an Artificial Neural Network Mechanism

The following derivation assumes a PID controller as a baseline controller. Let us consider the system to be controlled expressed as:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = f(x,u) \\ \dot{x}_3 = g(x,u) \end{cases} \tag{6-1}$$

where $x = [x_1^T \quad x_2^T \quad x_3^T]^T$ is the state vector, u is the input vector, x_2 is the controlled state vector, x_1 is the integral of the controlled states, and f and g are non-linear functions. Under AC, it can be assumed that the system becomes:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = f(x,u) + \Delta \\ \dot{x}_3 = g(x,u) + \bar{\Delta} \end{cases} \tag{6-2}$$

where f and g may now be regarded as approximations of the systems under ACs. Let the setpoint be expressed as x_{2ref}. Then, a baseline PID controller will produce a command u_{cmd} such that:

$$u = f(x,u) = u_{cmd} + \dot{x}_{2ref} \tag{6-3}$$

therefore:

$$f(x,u) = K_P\left(x_{2ref} - x_2\right) + K_I \int \left(x_{2ref} - x_2\right)dt + K_D\left(\dot{x}_{2ref} - \dot{x}_2\right) + \dot{x}_{2ref} \tag{6-4}$$

The tracking error can now be defined as:

$$e = \begin{bmatrix} x_{1ref} - x_1 \\ x_{2ref} - x_2 \end{bmatrix} \tag{6-5}$$

moreover, the tracking error dynamics are:

$$\dot{e} = \begin{bmatrix} \dot{x}_{1ref} - \dot{x}_1 \\ \dot{x}_{2ref} - \dot{x}_2 \end{bmatrix} \tag{6-6}$$

It can be immediately noted that:

$$\dot{x}_{1ref} - \dot{x}_1 = x_{2ref} - x_2 \tag{6-7}$$

For the second element of the error dynamics, let us consider the following:

$$\dot{x}_2 - \Delta = f(x,u) \tag{6-8}$$

then according to (6-4):

$$K_P\left(x_{2ref} - x_2\right) + K_I \int \left(x_{2ref} - x_2\right)dt + K_D\left(\dot{x}_{2ref} - \dot{x}_2\right) + \dot{x}_{2ref} - \dot{x}_2 + \Delta = 0 \tag{6-9}$$

or:

$$K_I\left(x_{1ref} - x_1\right) + K_P\left(x_{2ref} - x_2\right) + (K_D + I)\left(\dot{x}_{2ref} - \dot{x}_2\right) + \Delta = 0 \tag{6-10}$$

and:

$$\dot{x}_{2ref} - \dot{x}_2 = -(K_D + I)^{-1}\left[K_I\left(x_{1ref} - x_1\right) + K_P\left(x_{2ref} - x_2\right) + \Delta\right] \quad (6\text{-}11)$$

Therefore, the error dynamics can now be expressed as:

$$\dot{e} = \begin{bmatrix} x_{2ref} - x_2 \\ -(K_D + I)^{-1}K_I\left(x_{1ref} - x_1\right) - (K_D + I)^{-1}K_P\left(x_{2ref} - x_2\right) - (K_D + I)^{-1}\Delta \end{bmatrix} \quad (6\text{-}12)$$

$$\dot{e} = \begin{bmatrix} 0 & I \\ -(K_D + I)^{-1}K_I & -(K_D + I)^{-1}K_P \end{bmatrix}\begin{bmatrix} x_{1ref} - x_1 \\ x_{2ref} - x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -(K_D + I)^{-1} \end{bmatrix}\Delta \quad (6\text{-}13)$$

$$\dot{e} = \begin{bmatrix} 0 & I \\ -(K_D + I)^{-1}K_I & -(K_D + I)^{-1}K_P \end{bmatrix} e + \begin{bmatrix} 0 \\ -(K_D + I)^{-1} \end{bmatrix}\Delta \quad (6\text{-}14)$$

Under nominal conditions $\Delta = 0$ and K_P, K_I, and K_D can be obtained to ensure asymptotic stability for the error dynamics. For example, to control one state, x_1 and x_2 are scalars and so are K_P, K_I, and K_D. Therefore:

$$\dot{e} = \begin{bmatrix} 0 & I \\ -\dfrac{K_I}{K_D + I} & -\dfrac{K_P}{K_D + I} \end{bmatrix} e \quad (6\text{-}15)$$

The characteristic equation is:

$$\begin{vmatrix} s & 1 \\ -\dfrac{K_I}{K_D + I} & s + \dfrac{K_P}{K_D + I} \end{vmatrix} = 0 \quad (6\text{-}16)$$

or:

$$s^2 + \frac{K_P}{K_D + I}s + \frac{K_I}{K_D + I} = 0 \quad (6\text{-}17)$$

$$s_{1,2} = \frac{1}{2}\left(-\frac{K_P}{K_D + I} \pm \sqrt{\left(\frac{K_P}{K_D + I}\right)^2 - 4\frac{K_I}{K_D + I}}\right) \quad (6\text{-}18)$$

For stability: $\left(Re\left(s_{1,2}\right) < 0\right.$. Also note that: $2\zeta\omega_n = \frac{K_P}{K_D+I}$ and $\omega_n^2 = \frac{K_I}{K_D+I}$. For desirable values of ζ and ω_n one can solve for K_P, K_I, and K_D.

The ANN output is produced such that:

$$u = u_{cmd} - u_{ANN} \quad (6\text{-}19)$$

therefore:

$$\dot{e} = \begin{bmatrix} 0 & I \\ -(K_D + I)^{-1}K_I & -(K_D + I)^{-1}K_P \end{bmatrix} e + \begin{bmatrix} 0 \\ -(K_D + I)^{-1} \end{bmatrix} (\Delta - u_{ANN}) \qquad (6\text{-}20)$$

An on-line learning single hidden layer (SHL) ANN is selected due to its simple structure and demonstrated potential [101]. Figure 6-3 presents a schematic implementation of the SHL NN adaptive control mechanism.

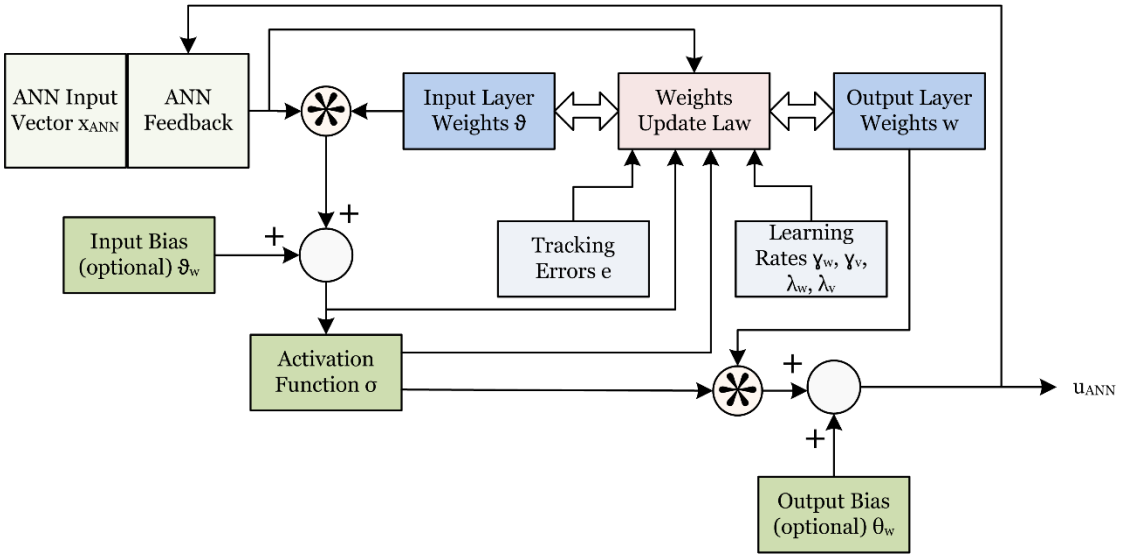

Figure 6-3. Single Hidden Layer ANN

Let us assume a general structure where we have n x_{ANN} inputs, p neurons in the hidden layer, and m outputs of the ANN. Any ANN output $u_{ANNi}, i = 1,2, \dots, m$ is computed using the following relationship:

$$u_{ANNi} = \sum_{j=1}^{p} \left[w_{ij} \sigma \left(\sum_{k=1}^{n} v_{jk} x_{ANNk} + \theta_{vj} \right) + \theta_{wi} \right] \qquad (6\text{-}21)$$

where w_{ij} are the interconnection weights between the hidden layer and the output layer, v_{jk} are the interconnection weights between the input layer and the hidden layer, and θ_{vj} and θ_{wi} are bias terms. The activation function is:

$$\sigma(\xi) = \frac{1}{1 + e^{-a\xi}} \qquad (6\text{-}22)$$

where the activation potential a is a design parameter. Note that the derivative $\dot{\sigma}$ of the activation function is:

$$\dot{\sigma}(\xi) = \frac{d\sigma}{d\xi} = a * \sigma(1 - \sigma) \tag{6-23}$$

The weights are updated according to the following update laws:

$$\begin{cases} \dot{w} = -\gamma_w[(\sigma - \dot{\sigma} * v^T x_{ANN})e^T + \lambda_w * \|e\| w] \\ \dot{v} = -\gamma_v[x_{ANN} * e^T * w^T * \sigma + \lambda_v * \|e\| v] \end{cases} \tag{6-24}$$

where e are state tracking errors and $\gamma_w, \gamma_v, \lambda_w, \lambda_v$ are learning rates (design parameters).

6.2. AIS-Based Adaptive Controller

The immunity-based adaptive mechanism mimics the humoral immune system feedback response. This represents the regulatory action of immunity specialized cells on the generation of antibodies [21]. The balance between the number and virulence of antigens, on the one hand, and the number and effectiveness of antibodies, on the other, is assessed by immunity cells and the generation of antibodies is accelerated or suppressed accordingly [22]. A simplified model of this process is implemented as an additional compensatory layer that increases or reduces the commands produced by baseline control laws with fixed parameters. The mechanism is equivalent to an adaptive modification of the controller gains when the operation of the system departs from nominal conditions. The architecture of the adaptive control laws using the immunity-based mechanism [102] to augment a baseline controller is presented in Figure 6-4.

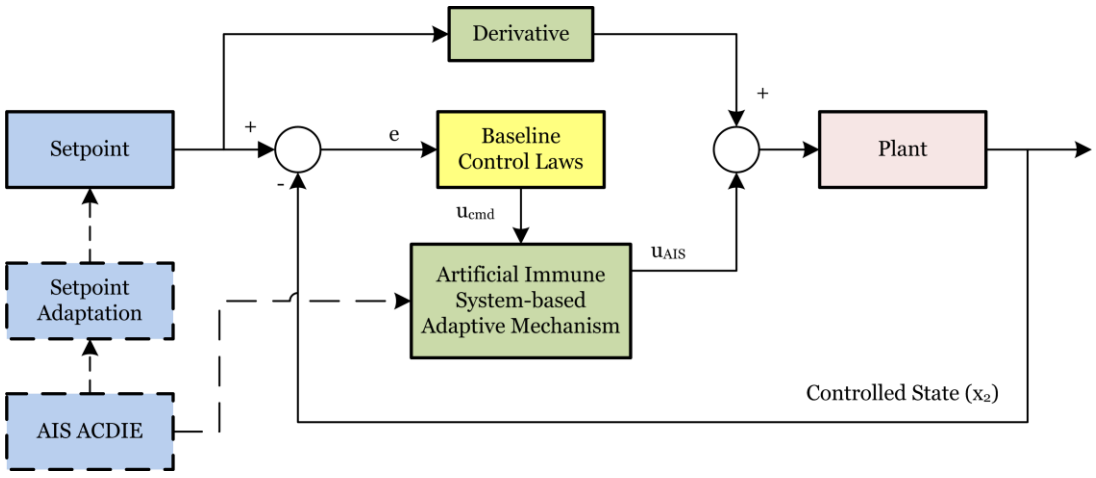

Figure 6-4. Adaptive Control Based on an AIS Mechanism

The antigens α_a active in triggering the immunity reaction are the result of the antiseptic action α_d of the antibodies on the invading antigens α. Therefore:

$$\alpha_a = \alpha - \alpha_d \tag{6-25}$$

The antibodies u, which are the main product of the immune system, are released in the blood stream and some of them (u_a) take an active role in locating destroying antigens, such that:

$$u_a = F_6(u) \tag{6-26}$$

and:

$$\alpha_d = F_7(u_a) \tag{6-27}$$

The active antigens trigger the excitation τ_e of mechanisms that produce helper T-cell τ_H, such that:

$$\tau_e = F_1(\alpha_a) \tag{6-28}$$

and:

$$\tau_H = F_2(\tau_e) \tag{6-29}$$

The number of helper T-cells reflects the number and virulence of the antigens in the organism and hence helper T-cells favor the generation of B-cells, which in turn accelerate the production of antibodies. Suppressor T-cells τ_S reflect the level of success of the immune system in counteracting the antigens. They are produced depending on the current amount of antibodies and the current amount and virulence of antigens:

$$\tau_S = F_5(\tau_e, u) \tag{6-30}$$

Suppressor T-cells are supposed to inhibit the production of B-cells and hence the production of antibodies such that a proper balance between the exogenous attack by antigens and the organism reaction is reached. In other words, the production of antibodies must be at the necessary level to defend the organisms but must not be excessive, such that resources are not wasted and other negative effects on the organism are avoided. Therefore, the production and activation of B-cell β is regulated by the balance between helper and suppressor T-cells:

$$\beta = F_3(\tau_B) \tag{6-31}$$

where:

$$\tau_B = \tau_H - \tau_S \tag{6-32}$$

The block diagram of this immunity mechanism is presented in Figure 6-5. It should be noted that the function F_1 may be interpreted as the action of the innate immune system (first line of defense, more or less indiscriminate destruction of outside invaders), while the rest of the block diagram may be interpreted as the action of the adaptive immune system.

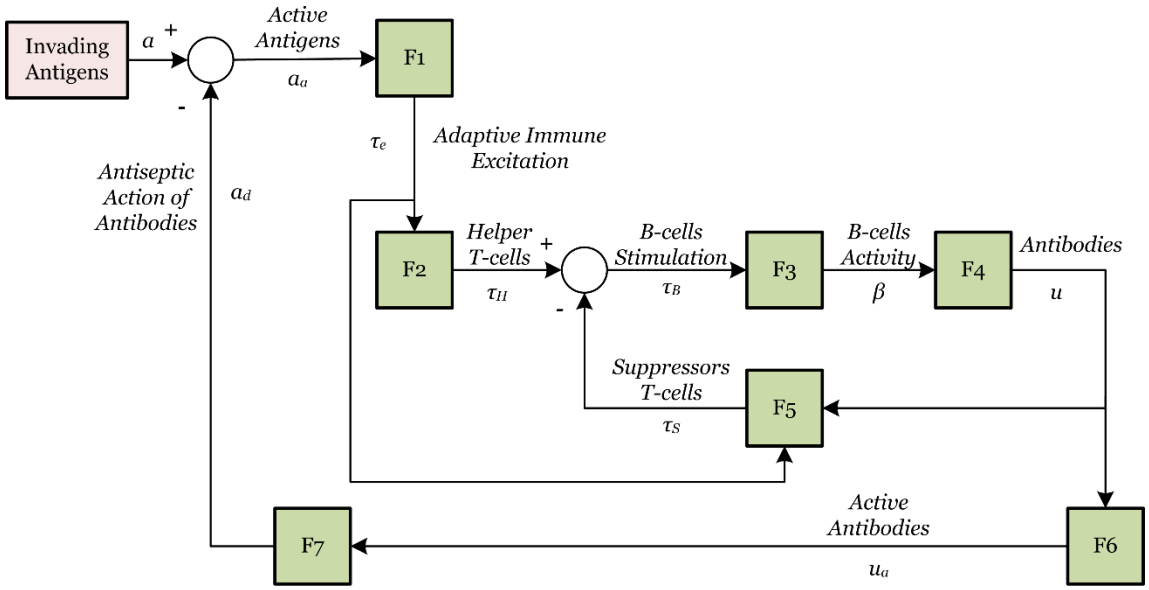

Figure 6-5. Humoral Immune System Feedback Mechanism

The biological immune system humoral feedback mechanism is a very complex process and all the actual mechanisms represented by F_1 through F_7 are not fully understood. Therefore there is still a lot of work to be done to come up with pertinent formulation of all these blocks. However, attempts have been made and what follows is a summary of what can be found in the literature.

F_1 is considered to be the identity function, for the biological system:

$$y = F_1(x) = x \tag{6-33}$$

If F_1 is interpreted as the action of the innate immune system (although previous researchers have not done that explicitly [76], [103]), when converting the biological mechanism to solve technical problems, this function becomes a PID (or P or PI) control law.

F_2 is considered to be the identity function [76], [103]. However, there are technical applications where F_2 is considered a different function [104]-[106], without clear justification:

$$F_2(x) = x \tag{6-34}$$

$$F_2(x) = \frac{1}{1 + e^{-kx}} \tag{6-35}$$

$$F_2(x) = \text{fuzzy logic function} \tag{6-36}$$

F_3 is considered to be an integral function on most of the reviewed papers [76], [103], [105] based on the idea that the activity of B-cells is the result of summing up the action of all B-cells produced.

F_4 is considered to be a derivative function [76], [103], [105] based on the idea that the production of antibodies is the result of the rate at which B-cells are generated. Therefore, for technical applications, the combined effect of F_3 and F_4 is considered to be just a gain, which results when the integral and derivative functions cancel each other.

F_5 is considered to have the following general form:

$$F_5(\tau_e, u) = k\tau_e f(u) \tag{6-37}$$

where *f(x)* used in ref [103], [76], [104] and [105] in this order are:

$$f(x) = 1 - e^{-\frac{x^2}{a}} \;;\; x = \dot{u}(t - \Delta t) \tag{6-38}$$

$$f(x) = 1 - \frac{2}{e^{ax^2} + e^{-ax^2}} \;;\; x = \dot{u}(t - \Delta t) \tag{6-39}$$

$$f(x) = 1 - \frac{2}{e^{ax} + e^{-ax}} \;;\; x = \dot{u}(t - \Delta t) \tag{6-40}$$

$$f(x) = FL[u(t - \Delta t), \dot{u}(t - \Delta t)] \tag{6-41}$$

where FL is a fuzzy logic function and $u(t - \Delta t)$ and $\dot{u}(t - \Delta t)$ are fuzzy inputs. It can be argued that this formulation captures the idea that suppressor T-cell will counteract helper T-cell when the predicted antibody generation is "large" and allow them to trigger antibody generation otherwise. The predicted antibody generation is represented by the rate at which u is produced, that is $\dot{u}$. A delay Δt is considered to count for the material

time needed for these processes to take place. When the rate of antibody generation is large, *f(x)* goes to 1 and the suppression action becomes maximum based on the assumption that enough antibodies will be generated. However, the same effect occurs if the rate of antibody generation is large negative. When the number of antibodies reaches some constant value (rate goes to zero), *f(x)* and the suppression action go to 0, allowing the generation of antibodies to be controlled mostly by the helper T-cells.

For a technical implementation, the invading antigens may be considered to be equivalent to the input to the system or the setpoint. This is based on the idea that the plant (assimilated to the organism or the blood stream in Figure 6-5 must "match" the setpoint with the actual value of the controlled variable. Therefore:

$$F_6(u) = controlled\ plant \tag{6-42}$$

To include the material time necessary for the antibodies to locate and actually destroy antigens, F_7 represents a time delay and a proportionality factor:

$$F_7(u_a) = ku_a(t - \Delta t) \tag{6-43}$$

Note that the delay in (6-43) and/or in (6-38)-(6-41) is often neglected in practical implementations.

The functions considered are presented next and the final immunity based adaptive controller architecture is presented in Figure 6-6. F_B is the function f in equations (6-38)-(6-41) above or Eqn. (37) from the paper by Perez et al [76]. This is equivalent to:

$$F_1 = PID \tag{6-44}$$

$$F_2 = identity \tag{6-45}$$

$$F_3F_4 = K \tag{6-46}$$

$$F_5 = k\eta f(u) \tag{6-47}$$

$$F_6(u_t) = controlled\ plant \tag{6-48}$$

$$F_7 = identity \tag{6-49}$$

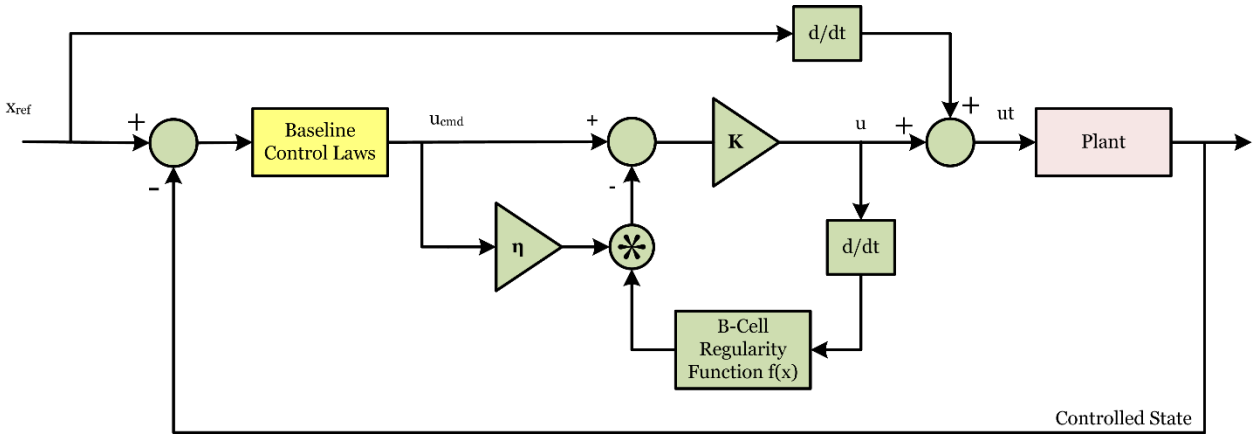

Figure 6-6. AIS-based Adaptive Mechanism

Chapter 7. AIS Based Evolutionary Optimization

Power plant control requires proper control system architecture, possibly variable/adaptive setpoints (i.e. setpoint optimization), and adequate control system parameters (i.e. controller gain optimization) for optimal operation. Once the control system architecture is established, the remaining two problems are generally characterized by strong nonlinearities, multi-dimensionality, the existence of multiple local extrema, and various constraints. Evolutionary optimization algorithms have been demonstrated to provide the needed capabilities for solving such problems. This chapter presents the development of an interactive computational environment for the optimization of power plant control using evolutionary techniques with immunity-inspired enhancements.

7.1. The Genetic Algorithm

Genetic algorithms (GA) are artificial intelligence techniques inspired by the biological species evolution theory [107], [108] that are implemented for parameter or function optimization. In biological evolution theory, individuals within a population that are more fit to a given environment are more likely to survive long enough to produce offspring, while unfit individuals are more likely to die off before they produce offspring. When an individual produces offspring, many of the characteristic that facilitated its survival are passed down to its offspring. Thus, over many generations, through the mechanism of natural selection, the fitness of the individuals within a population is expected to increase, eventually reaching an optimum.

In biological organisms, deoxyribonucleic acid (DNA) serves as a type of map that defines the traits, characteristics, and inner workings of the organism. Within DNA, genes contain instructions used for organisms' development and reproduction. A chromosome is a threadlike linear strand of genetic material, or genes. When two organisms produce an offspring, each parent passes on a portion of their DNA to the child; the DNA of the offspring is a combination of parent's DNA. During cell division, sections of genes from one chromosome may be swapped with sections of genes from another chromosome; this is referred to as crossover. In addition to crossover, random mutations may also occur and alter individual genes. Through the processes of crossover and mutation, the offspring's genetic material may differ from that of either parent, and thus will express different character traits which may make the individual more or less suited to a given

environment. Over many generations, the random variations in offspring and natural selection mechanisms lead to individuals better suited for survival in a given environment than were originally present in earlier populations.

7.2. The Genetic Algorithm Description

When applying the GA to solve an optimization problem, an individual is defined as a potential solution. Within an individual, a gene is used to refer to a particular solution parameter. The population is defined as a collection of individuals. A genetic operator is an action that results in a modification to an existing individual's gene configuration.

A summary of the evolutionary optimization process is as follows. As in the biological analogy, an initial population must exist. The initial population is generated randomly within pre-determined bounds. The initial population is then rated using a performance index function, which serves the purpose of the environment. A new population is generated based on the fitness of the individuals in a process that mimics natural selection. The individuals in the new population are subjected to random mutations and crossover operations. The evolutionary operations repeat until a convergence criterion is met, generally until a performance index has met or until a predefined number of generations reached. The distinct stages of this algorithm will be explained in more detail in the following section. An overview of the GA optimization algorithm is presented in Figure 7-1.

7.2.1. Initial Population

An integer representation of candidate solutions, as a variation of the binary representation, has been adopted. This solution allows for a more localized mutation operator, which is expected to provide a better balance between exploration and exploitation within the cloning process described later in this chapter.

For the controller gain optimization, a candidate solution or a chromosome consists of a set of values for the gains, which may be regarded as genes. A total number N_g of real-valued gains $(g_q \in \mathcal{R}, q = 1,2,\dots,N_g)$ must be specified. Other required parameters are the ranges of the gains such that:

$$g_q \in [g_{qmin}, g_{qmax}], q = 1,2,\dots,N_g \tag{7-1}$$

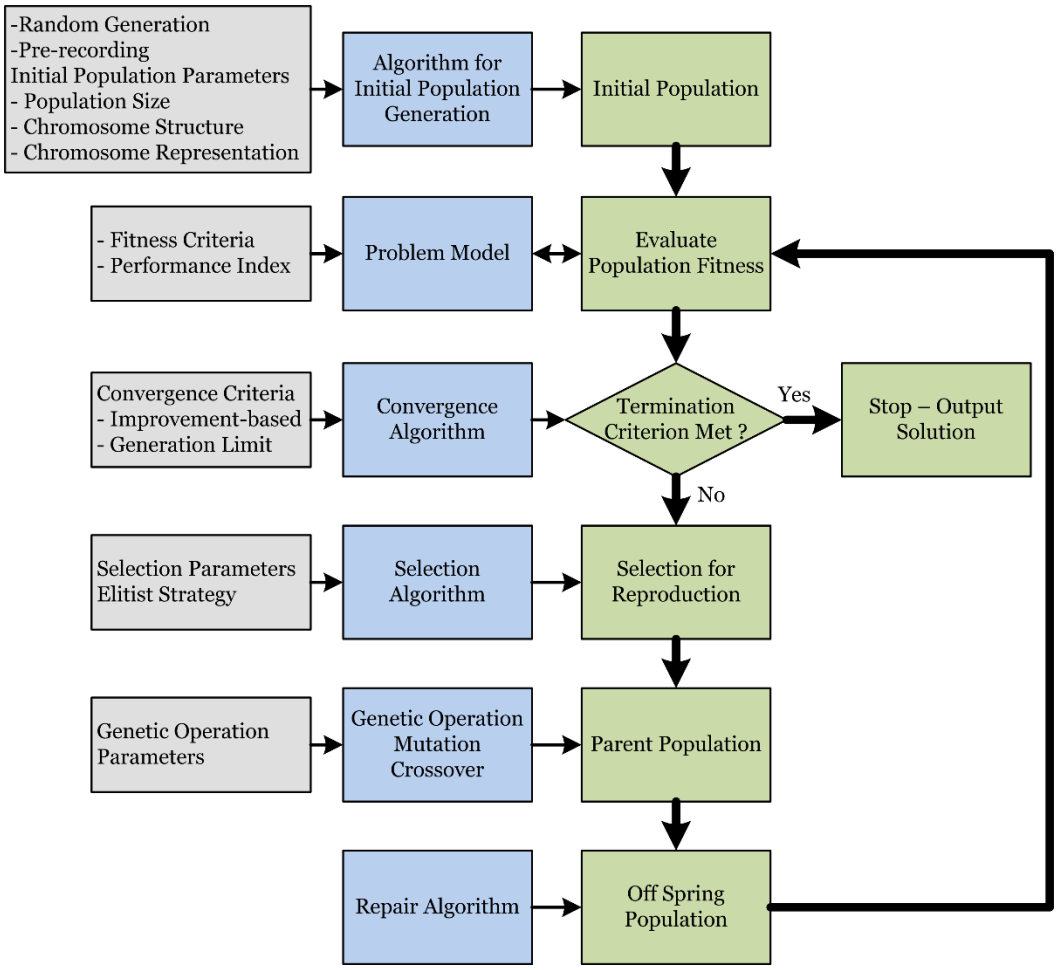

Figure 7-1. Overview of the GA

Moreover, the integer resolution $gres_q \in I$ of the gain search space, representing the size of the gain range partition. The topology of the representation is illustrated in Figure 7-2. Note that the integers representing the genes are defined as:

$$integer\ \#q = \left\lceil \frac{g_q - g_{qmin}}{g_{qmax} - g_{qmin}} * gres_q \right\rceil \tag{7-2}$$

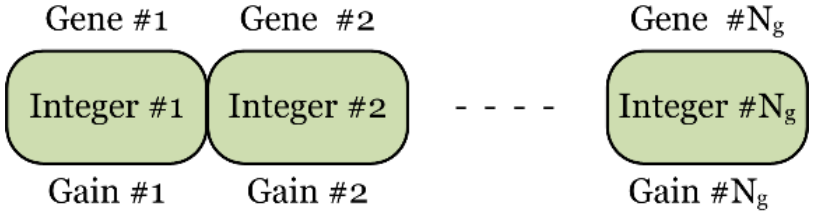

Figure 7-2. Chromosome Structure for Gain Optimization

For the setpoint optimization problem, a solution represents a set of setpoint values maintained constant over time intervals of various duration. For this case, the

required parameters include: input cycle duration (t_c), maximum number of setpoint intervals (N_{smax}), setpoint range, and setpoint resolution. Note that the number of setpoint intervals N_s for each individual and the start time t_{sj} of each interval j are optimization objectives, such that:

$$1 \leq N_s \leq N_{smax} \tag{7-3}$$

and:

$$t_{s1} \leq t_{s2} \leq \cdots \leq t_{sN_s} \leq t_c \tag{7-4}$$

The structure of the chromosome for this optimization problem is illustrated in Figure 7-3. A total number N_g of real-valued setpoints ($sp_q \in \mathcal{R}, q = 1,2, \dots, N_g$) must be specified. Other required parameters are the ranges of the setpoints such that:

$$sp_q \in [sp_{qmin}, sp_{qmax}], q = 1,2, \dots, N_g \tag{7-5}$$

Moreover, the integer resolution $spres_q \in I$ of the gain search space, representing the size of the setpoint range partition. Note that the integers representing setpoint integers are defined such that:

$$integer\ \#2q = \left\lceil \frac{sp_q - sp_{qmin}}{sp_{qmax} - sp_{qmin}} * spres_q \right\rceil \tag{7-6}$$

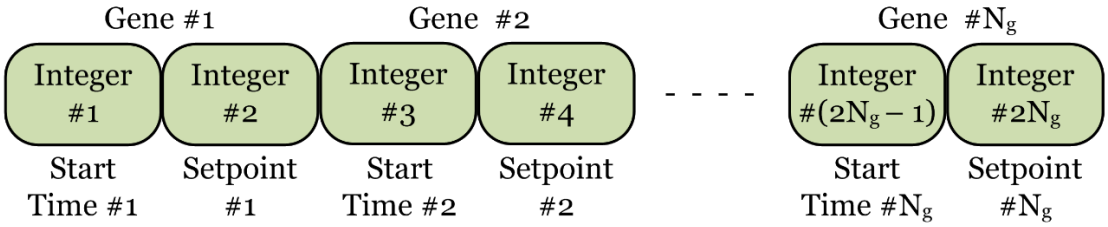

Figure 7-3. Chromosome Structure for Setpoint Optimization

The initial population is defined as a N_I by N_g matrix of N_I individuals containing N_g genes. There exit different approaches to generate the initial population; however, only two approaches are covered here. The first method is to generate random values for each gain/parameter that lie between the pre-defined lower and upper bounds. This approach has the benefit of providing more variety to the population, which leads to a more thoroughly explored solution space. The disadvantage of this method is that the high

dimensionality of the chromosomes and the complexity of the controller could lead to very poor starting parameters; thus the GA will take much longer to converge.

7.2.2. Fitness Function

The fitness function is used by the GA to evaluate how well a given individual (i.e. solution) archives the optimization criteria. Any evaluating metric can be utilized as long as it rewards desired performance. The fitness function relies on the establishment of a set of N_p performance criteria and associated numerical metrics. The level of attainment of each criterion in terms of the associated metrics must be determined through simulation. These values are scaled between 0 and 1 (with 1 being the best and 0 the worst) to produce elementary performance or fitness evaluations $pe_i, i = 1,2, \dots, N_p$. The overall fitness function FF of a potential solution is then defined as the weighted average of all elementary performance evaluations as described in section 3.5.

For the controller optimization problem, the set of a possible performance criteria used for the numerical example presented in this dissertation includes the following:

- Rise time, the time required for the process controlled variable to go from 10% to 90% of the desired steady-state set point
- The maximum of the absolute value of the tracking error calculated after rise time
- The mean of the tracking error
- The standard deviation of the tracking error
- The integral of the absolute value of the tracking error

For the setpoint optimization problem, general process performance criteria may be considered such as maximum or imposed amount of H_2S and CO_2, minimum amount of solvent, minimum total cost, and minimum environmental impact.

7.2.3. Parent Population Selection

Selection of the parent population for the next generation is performed using the roulette-wheel selection approach [109]. In this approach, each individual is given a survival rate based on its performance relative to the total performance of the population. The total performance index TF is the sum of all of the performance indices for all individuals in the current population, such that:

$$TF = \sum_{i=1}^{N_I} PI_i \tag{7-7}$$

The performance index of each individual is then divided by the total performance index TF of the current population to obtain the probability of selection for each individual $\overline{PI}_i$, such that:

$$\overline{PI}_i = \frac{PI_i}{TF}, i = 1,2, \dots, N_I \tag{7-8}$$

Finally, the cumulative probability QP_i is calculated next for each of the individuals, as the sum of the probabilities of all precedent individuals such that:

$$QP_i = \sum_{j=1}^{i} \overline{PI}_i \tag{7-9}$$

Each available spot in the new population is filled by generating a random number in the close interval [0, 1] and selecting an individual in which the random number is less than its cumulative probability, but greater than the cumulative probability of the preceding individual. Individuals with higher performance indices will get larger cumulative probability intervals and therefore more chances for survival and multiple copies in the parent generation. Since the population size is maintained constant throughout the algorithm, the population in each generation can only contain the same number of individuals as in the initial population. The algorithm continues until the next generation is fully populated. Figure 7-4 present a flowchart of the roulette-wheel selection approach.

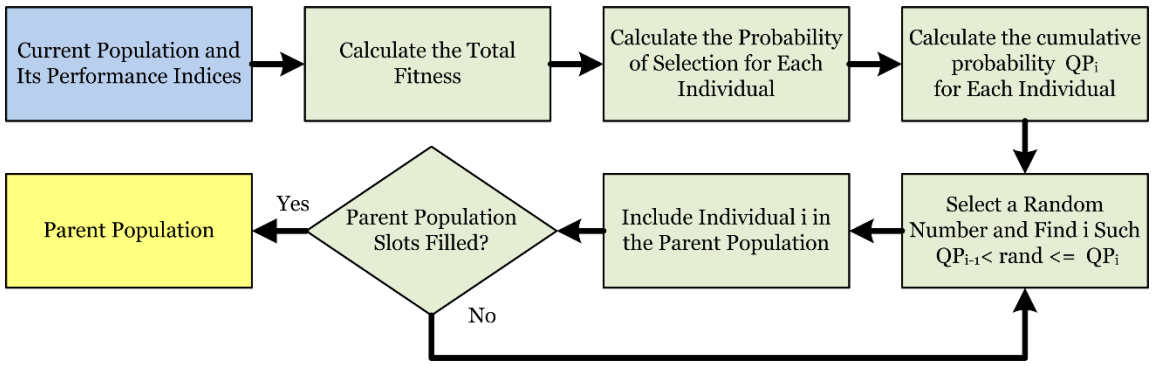

Figure 7-4. Roulette-Wheel Selection Approach

It is worth mentioning that the elitist strategy has been used after the parent population has been selected. The strategy ensures the best current solution is not lost in the selection process. The elitist strategy is performed by replacing a randomly selected individual in the newly generated population with the best current solution available from the previous generation.

7.2.4. Mutation

The first genetic operation performed on the population is mutation. First, the mutation rate (ν) is defined as the percentage of the genes in the population that should statistically be subjected to a mutation operation. Next, a random decision matrix (Λ) is defined in Eq. (7-10). In this equation, the rand operator produces a N_I by N_g matrix of uniformly-distributed random numbers in the closed interval from 0 to 1.

$$\Lambda = rand(N_I, N_g) \tag{7-10}$$

Next, an N_I by N_g matrix of genes (X) are chosen for mutation is generated by:

$$\mathrm{X}_{iq} = \begin{cases} 1 & if\ \Lambda_{iq} \leq \nu \\ 0 & if\ \Lambda_{iq} > \nu \end{cases} \tag{7-11}$$

In this equation, the subscript i represents the index of individual in the population, and subscript q represents the index of the gene.

An N_I by N_g chromosomal modification limit matrix Ψ is defined next:

$$\Psi = \mathrm{ones}(N_I, 1) * \sigma \tag{7-12}$$

In the above equation, the ones operator is an N_I by 1 vector where each element is equal to 1. The chromosomal modification limit (σ) is defined as a 1 by N_g vector. The indices of the modification limit vector define the maximum absolute value by which a gene selected for mutation can mutate.

Finally the mutation modification matrix (Υ) is defined as:

$$\Upsilon = -\Psi + 2 * rand(N_I, N_g) \otimes \Psi \tag{7-13}$$

where $\otimes$ is used to indicate a piecewise matrix multiplication.

Once the potential modification matrix is calculated, it can be used to produce the new mutated population P_m from the current population P_c, such that:

$$P_m = [P_c + \Upsilon \otimes X] \tag{7-14}$$

A flowchart of the mutation process is presented in Figure 7-5.

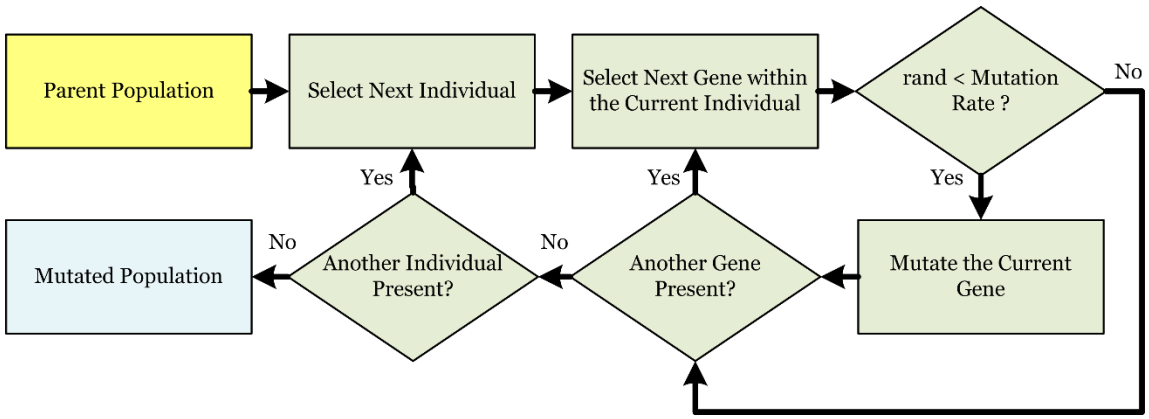

Figure 7-5. The Mutation Operation

7.2.5. Crossover

To perform the crossover operation, the crossover rate (c) must be defined. The crossover rate represents half of the probabilistic percentage of the population that should be subjected to a crossover operation in one generation. Note that this is defined as half because each crossover operation affects two individuals. For each individual in the population, a random number in the range [0, 1] is chosen. If c is larger than this random number, then a crossover operation is performed on the current individual.

To perform a single point crossover operation on the current individual I_c, a single crossover point integer index (a) is generated in the range$(1, N_g - 1)$, where N_g is the chromosome gene length. A second individual I_r is randomly selected from the population for crossover. The genes separated by the single crossover point are swapped among the two chromosomes as presented in Eq.(7-15), where the notation $I(e{:}\, f)$ is used to define a vector consisting of elements e through and including element f for the vector I. An illustration of the single point crossover operation is presented in Figure 7-6. The flowchart of the single point crossover algorithm is presented in Figure 7-7.

$$\bar{I}_c = \left[I_c(1{:}\, a) \;\; I_r(a+1{:}\; N_g) \right] \tag{7-15}$$

$$\bar{I}_r = [\, I_r(1{:}\,a)\;\; I_c(a+1{:}\;\; N_g)]$$

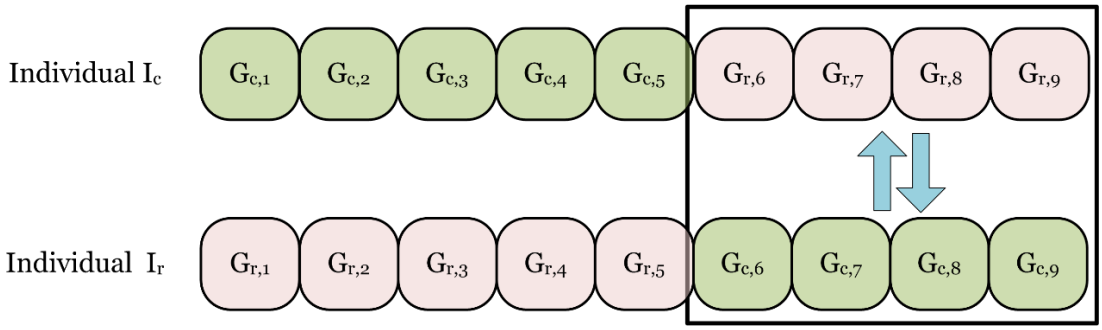

Figure 7-6. The Single Point Cross Operation

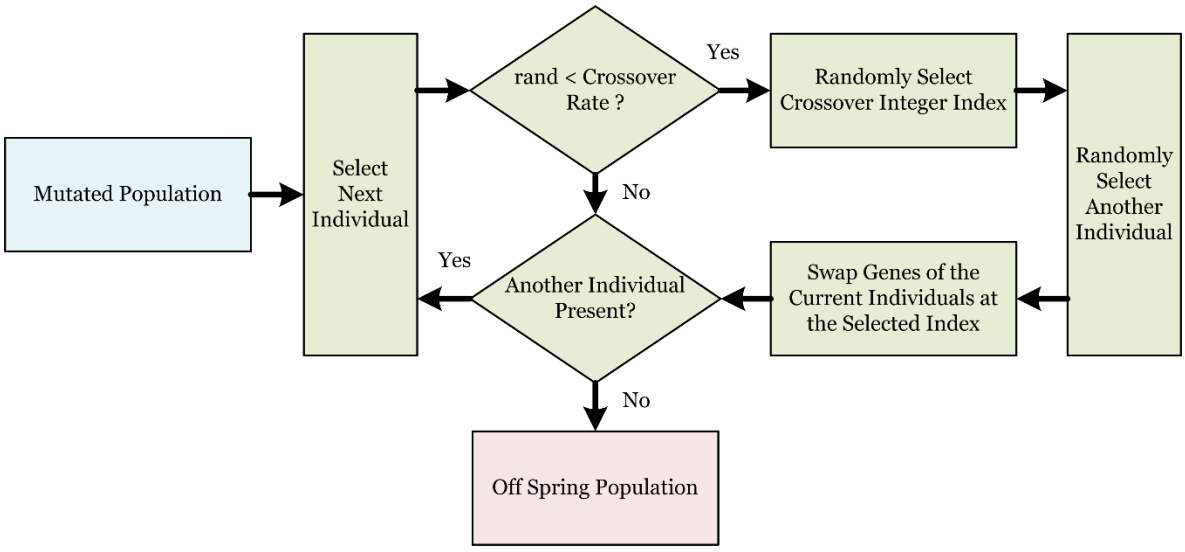

Figure 7-7. The Single Point Cross Algorithm

7.3. Immunity-Enhanced Genetic Algorithm

Immunity-inspired enhancements (Figure 7-8) include seeding and vaccination of initial population, clonal selection loop, and population diversification and selection based on affinity to self/nonself. Seeding and vaccination are expected to enhance the initial population fitness, the clonal loop is expected to enhance exploitation, while affinity-based selection is designed to improve exploration. Overall, these algorithms are expected to enhance the computational effectiveness, improve convergence, be more efficient in handling multiple local extrema, and achieve adequate balance between exploration and exploitation [110]. An interactive computational environment has been developed to facilitate the test and future development as descried in section 10.3.

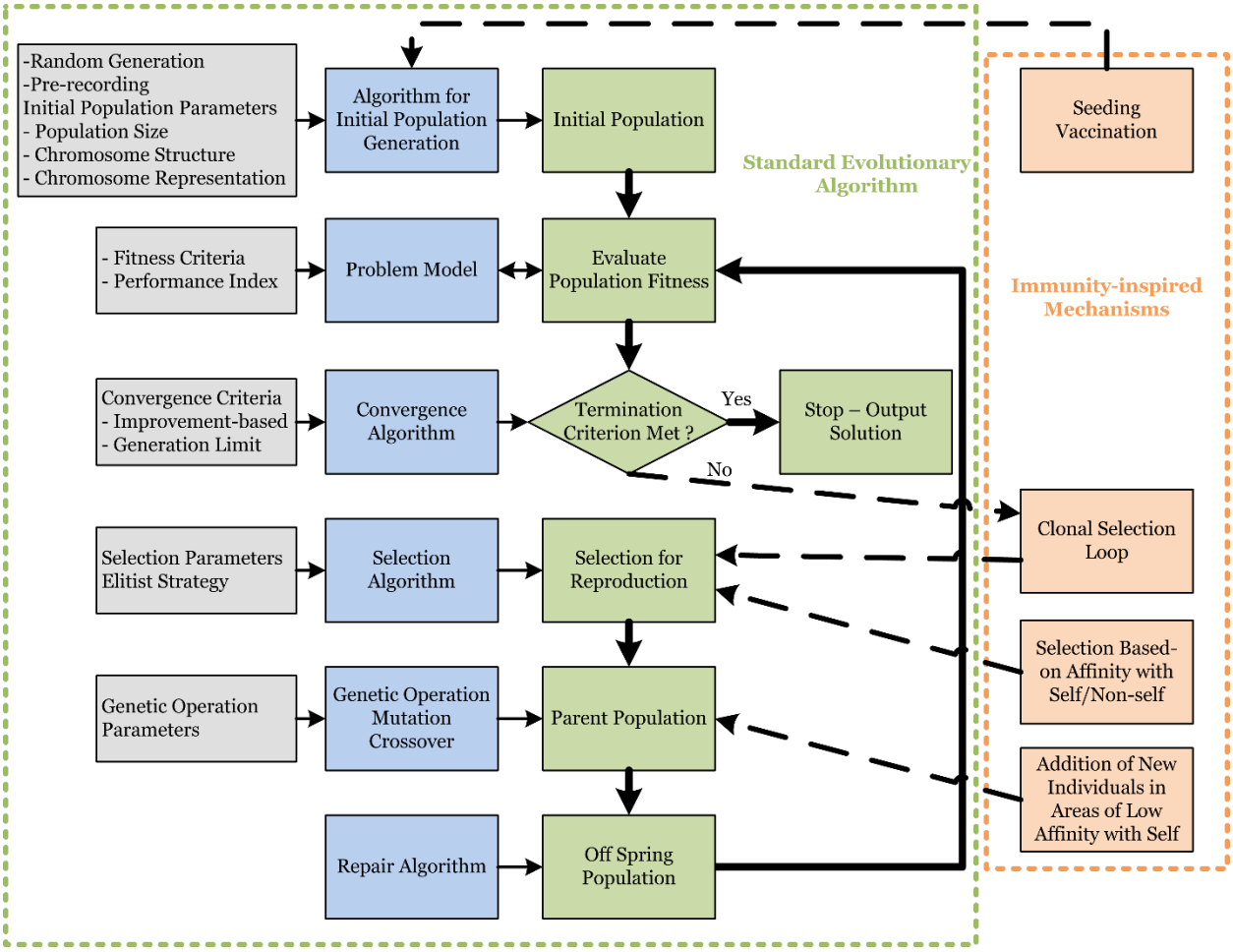

Figure 7-8. Immunity-Enhanced Genetic Algorithm Implementation

7.3.1. Seeding and Vaccination

The immunity-based enhanced algorithm uses seeding and/or vaccination in the generation of the initial population. They rely on previous knowledge on "good" solutions and are similar to the more rapid defense reaction of the immune system in response to previously experienced attacks. Seeding consists of introducing in the initial population complete pre-determined solutions, while vaccination consists of introducing partial solutions expected to achieve a high performance index. The block diagram of this process is presented in Figure 7-9. In this approach, certain percentage of the initial population is randomly generated, while the remaining percentage is introduced using the seeding and/or vaccination. The ratio of the number of the individuals introduced using the seeding and/or vaccination to the number of randomly generated individuals is very important. For instant, a higher ratio will emphasize the seeding and vaccination generated individuals and therefore enhance exploitation. A smaller ratio will be

providing more variety to the initial population rather than relying on already acquired information and thus enhancing exploration.

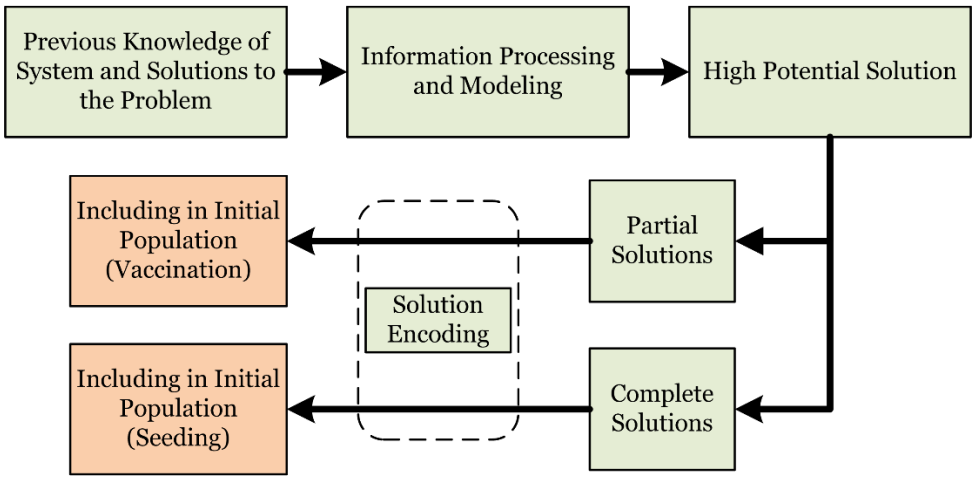

Figure 7-9. Seeding and Vaccination of Initial Population

7.3.2. Clonal Selection Loop

The clonal selection loop is inspired by the biological immune system T and B-cells. When receptors on the surface of a T-cell or a B-cell bind to an antigen, this cell gets stimulated to undergo proliferation and differentiation in a process that is called clonal selection. In the clonal selection loop, presented in Figure 7-10, exploitation is enhanced by producing multiple copies of each individual and exposing them to a hyper-mutation operator for several iterations at the end of which the best clone is selected into the parent set for the next generation.

The clonal selection loop is inserted after the current population evaluation and new generation selection in the standard genetic algorithm. Below is a description of the clonal loop functions and parameters.

1. **The Triggering Mechanism:** The clonal loop could be triggered using different mechanisms. For instant, a generation based triggering algorithm could be used to go through the loop after a certain number of generations. On the other hand, improvement-based triggering mechanism could be used to run the loop when the GA has a slow improvement rates over a pre-defined numbers of generations.

2. **The Cloning:** The cloning consist of generating a number of copies of each individual in the current population. The number of clones per individual or the total

number of clones must be specified. Each individual could get a constant number of clones or a number of clones relative to its performance index.

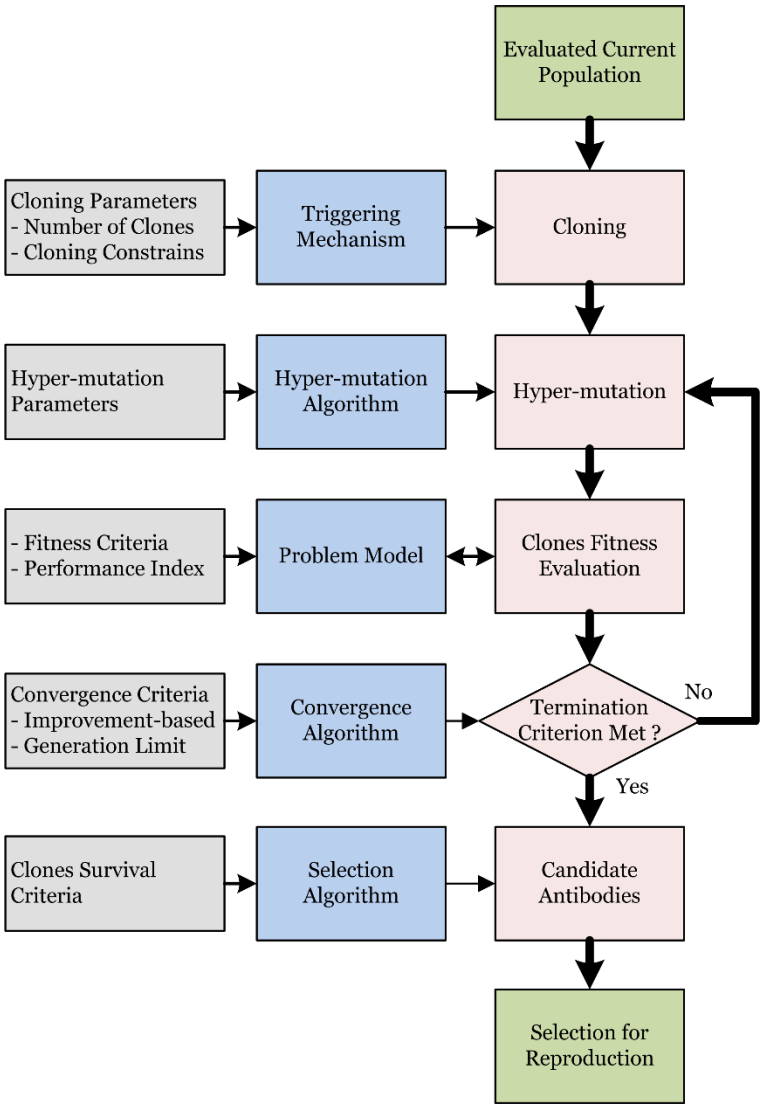

Figure 7-10. Clonal Selection Loop

3. **The Hyper-mutation:** The hyper-mutation consists of mutating the generated clones using a high mutation rate. The mutation algorithm described for the GA algorithm could be used for this purpose.

4. **Clones Fitness Evaluation:** Each generated clone must be evaluated using the GA fitness function.

5. **Convergence Criterion:** Two convergence criteria could be used to exit the clonal loop. The improvement-based approach will stop the algorithm when the percentage improvement in the best clones over a pre-defined number of iterations has

been reached. An imposed maximum number of iterations could also be used to exit the loop. A combination of two approaches is also possible.

6. **Candidate Antibodies:** Once the convergence criterion has been met, the best clone form each individual set of clones is kept to form the new population.

7.3.3. New Individuals Addition

For an adequate balance between exploitation and exploration through individual diversity, a number of new solutions may be introduced into the population based on their low affinity to the self. Vaccination and/or seeding algorithms, similar to those used for the initial population may be used in the construction of these new individuals.

Within the negative selection concept, cells that recognize the intruders are entities expected to "solve the problem". Therefore, the candidate solutions that achieve high performance indexes are assimilated to antibodies and will be referred to as "non-self". The individuals that do not achieve high performance indices are eventually undesirable and will be assimilated to the "self". During the evolutionary process, libraries of candidate solutions Ω_i and $\overline{\Omega}_i$ that belong to the self (L_s) and the non-self $(\overline{L}_s)$, respectively, are recorded and stored such that:

$$L_s = [\Omega_1 \ \ \Omega_2 \ \ \dots \ \Omega_{ns}\ ] \tag{7-16}$$

$$\overline{L}_s = [\overline{\Omega}_1 \ \ \overline{\Omega}_2 \ \ \dots \ \overline{\Omega}_{ns}\ ] \tag{7-17}$$

For two individuals C_1 and C_2 of the same size N_g, their relative affinity $\mathrm{A}\ (C_1,\ C_2)$ is defined as:

$$A(C_1, C_2) = [ones]\ .\ [|C_1 - \ C_2|]^T \tag{7-18}$$

where $[ones]$ represents a row vector with all N_g, components equal to 1. For a candidate solution C_i , the affinity to the self is defined as:

$$A(C_i, L_s) = \min_j [A(C_i, \Omega_j)] \tag{7-19}$$

On the other hand, the affinity to the non-self is defined as:

$$A(C_i, \overline{L}_s) = \min_j [A(C_i, \overline{\Omega}_j)] \tag{7-20}$$

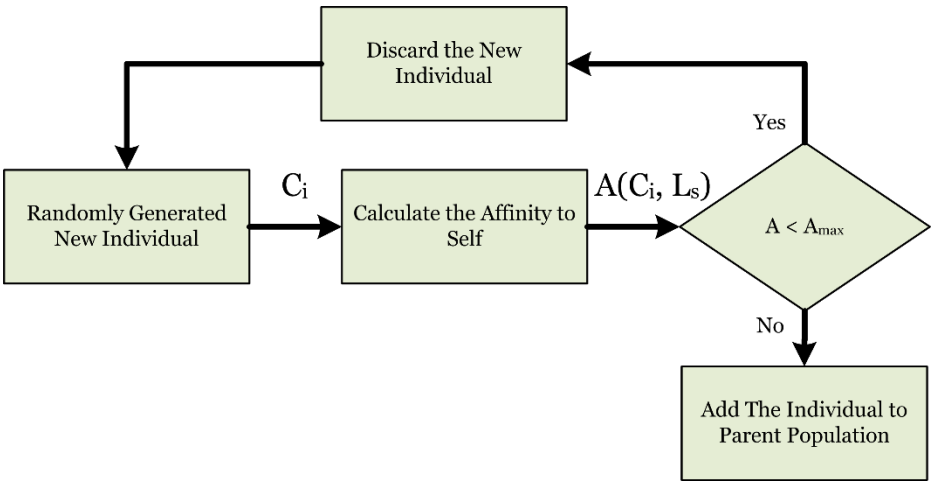

Figure 7-11. New Individual Generation Based on Low Affinity to Self

7.3.4. Affinity-based Selection Algorithm

An affinity-based selection algorithm is formulated as a modified roulette selection technique. The premise is that high performance solutions that have low affinity with respect to both the self and non-self should be investigated with priority. A fitness alteration factor Φ_i is defined as:

$$\Phi_i = \mathfrak{F}\left(A(C_i, L_s), A\left(C_i, \overline{L}_s\right)\right) \tag{7-21}$$

such that the fitness of each solution can be expressed as:

$$FF_i = \Phi_i . PW^T . PP_i \tag{7-22}$$

For this investigation, the function $\mathfrak{F}$ is defined as:

$$\mathfrak{F} = \begin{cases} 0.9 & if\ A(C_i, L_s) > A_{max}\ and\ A\left(C_i, \overline{L}_s\right) > \overline{A}_{max} \\ 1.2 & if\ A(C_i, L_s) < A_{max}\ and\ A\left(C_i, \overline{L}_s\right) < \overline{A}_{max} \\ 1 & otherwise \end{cases} \tag{7-23}$$

The flowchart of the proposed affinity-based selection algorithm is presented in Figure 7-12. New individuals can be generated at each iteration to enhance exploration and ensure diversity of the population as described in the previous section. It is desirable that these new solutions have low affinity to the self. In other words, they are located in regions other than those that have already been tested to produce low performance individuals. It should be noted that the non-self library can be used to produce vaccines

by identifying strings of genes that occur more frequently and are likely to lead to good performance individuals.

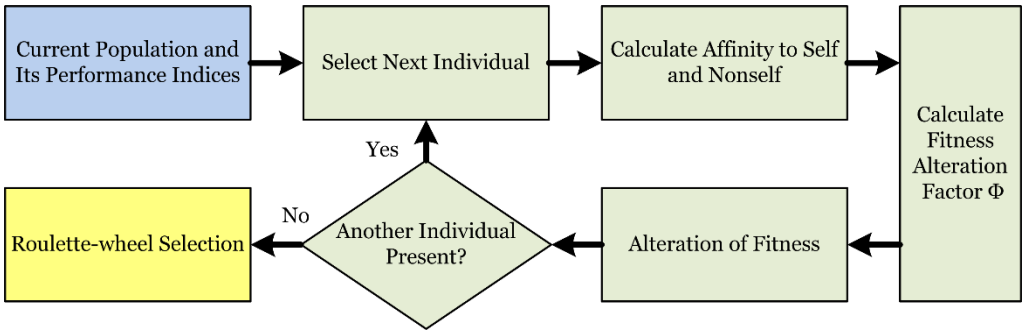

Figure 7-12. Affinity-based Individual Selection

Chapter 8. The Acid Gas Removal Unit

The acid gas removal (AGR) [111] unit is the targeted demonstration system in this study. The unit is part of the integrated gasification combined cycle (IGCC) power plant. IGCC is emerging as an attractive technology option for providing clean, low-cost energy with lowest carbon dioxide emissions among other coal power plants. The unit is highly non-linear, has thousands of features, tens of baseline controllers and could undergo several ACs, which make the unit a perfect candidate to test all the proposed framework algorithms.

8.1. The Acid Gas Removal Unit Description

The unit is expected to selectively remove hydrogen sulfides (H_2S) and carbon dioxide (CO_2) from the raw syngas using the physical solvent SELEXOL. SELEXOL solvent has an affinity to absorb both H_2S and CO_2 at high partial pressures and low temperatures and will release those gases when the solvent is depressurized and heated. The AGR is designed to produce clean syngas with less than 1 ppm H_2S in it and capture at least 95% of CO_2 to produce three products streams: clean syngas (to be used by the combustion turbine), H_2S gas (to be used in Clause Plant), and CO_2 gas (for compression and storage).

For the purpose of this study, the AGR unit has been divided into 22 subsystems and components listed in Table 8-1 including trayed and packed distillation columns, pressure vessels, heat exchangers, pumps, compressors, and strainers.

Table 8-1. The AGR Unit Subsystems

#	Subsystem Name	#	Subsystem Name
1	H_2S Absorber	12	Stripped Gas Knock Out Drum
2	CO_2 Absorber	13	Acid Gas Knock Out Drum
3	H_2S Concentrator	14	High Pressure Flash Drum
4	SELEXOL Stripper	15	Medium Pressure Flash Drum
5	Lean/Rich Solvent Heat Exchanger	16	Low Pressure Flash Drum
6	Lean Solvent Cooler	17	Stripped Gas Compressor
7	Loaded Solvent Cooler	18	Loaded Solvent Pumps
8	H_2 Recovery System	19	Lean Solvent Pumps
9	Reboiler	20	Semi-lean Solvent Pumps
10	Stripped Gas Cooler	21	Rich Solvent Strainer
11	Acid Gas Cooler	22	Lean Solvent Strainer

A schematic drawing of the AGR unit is presented in Figure 8-1 [112]. Section 8.1.1 through section 8.1.4 describe the major processes in the AGR unit: the H_2S absorber, CO_2 absorber, H_2S concentrator and H_2S stripper.

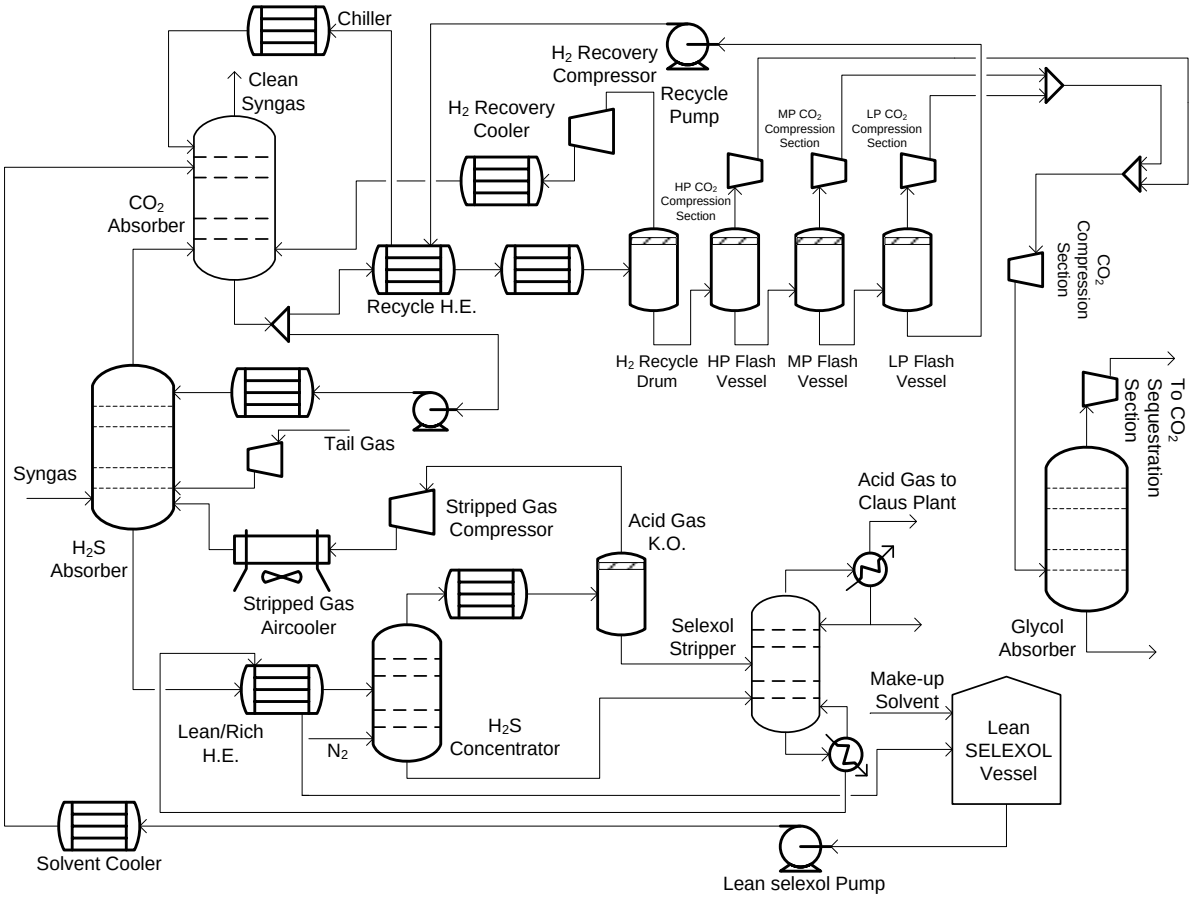

Figure 8-1. Schematic of the AGR Unit [112]

8.1.1. The H_2S Absorber

The raw syngas is cooled in the H_2S absorber feed cooler before it is fed into the lower section of the H_2S absorber along with the stripped gas coming from the stripped gas compressor. The semi-clean syngas leaves the top of the H_2S absorber toward the bottom of the CO_2 absorber. "Loaded" solvent pumped from the lower part of the CO_2 absorber is cooled to about 40°F in the loaded solvent coolers and fed to the upper section of the H_2S absorber. The loaded solvent flows downward across random packing and absorbs H_2S in the syngas and becomes "rich" solvent and flows from the bottom of the H_2S absorber. The rich solvent is then heated by lean solvent coming from the SELEXOL stripper as it flows through the lean/rich heat exchanger. The hot rich solvent then flows

to the upper of the H_2S concentrator, where it is first partially flashed via depressurization and then back-stripped with a slipstream of treated syngas. This step enriches the concentration of the H_2S in the H_2S stream going to the Claus Plant for further processing. The stripped gases from the H_2S concentrator are cooled in the stripped gas cooler, dried in the stripped gas knock out drum, compressed in the stripped gas compressor and sent back to the raw syngas inlet of the H_2S absorber. The solvent leaving the bottom of the H_2S concentrator is forwarded to the H_2S stripper to further free the remaining gases in the rich solvent.

8.1.2. The H_2S Stripper

Rich solvent from the H_2S concentrator enters the upper quadrant of the H_2S stripper at the top. The separated H_2S vaporizes and goes through the reflux trays and off the top the H_2S stripper, along with the stripping steam and a small portion of the solvent. This acid gas stream is then air cooled in the acid gas cooler, condensed liquids are removed from the acid gas stream, then collected in the acid gas knock out drum and pumped back to the H_2S stripper again as reflux and flown downward across the reflux trays to minimize SELEXOL solvent entrainment. A small portion of the reflux flow will be optionally required as blown down to keep the water content of the solvent constant. The acid gas exits from the acid gas knock out drum and leaves the AGR unit for processing in the Claus Plant. The solvent in the H_2S stripper is heated in the reboiler. The reboiler uses medium pressure steam to heat the solvent as it proceeds down the column, to generate stripping steam, and to provide the heat of desorption for H_2S and CO_2 from the solvent. The stripping steam rises in the column and aids in the separation of the incoming rich solvent until the H_2S exits at the top.

The lean solvent leaving at the bottom of the H_2S stripper is discharged using lean solvent pumps. The lean solvent is then cooled in lean/rich heat exchanger, as it heats the rich solvent flowing to the H_2S concentrator. The lean solvent then flows through the lean solvent cooler where it is further cooled before entering the CO_2 absorber.

8.1.3. The CO_2 Absorber

The syngas leaving the top of the H_2S absorber is fed to the bottom of the CO_2 absorber. At the same time, the lean solvent is fed to the upper quadrant of the CO_2 absorber, while the semi-lean solvent is fed in the middle of the absorber. Both lean and semi-lean solvents flow down over the top random packing and absorb the CO_2 in the

syngas. The clean syngas, which contains less than 1 ppm H_2S and some CO_2, leaves from the CO_2 absorber top.

The loaded solvent flows from the bottom of the CO_2 absorber and is separated into two streams: a significant portion is routed to the process of isolation of the captured CO_2 and the solvent. The reminder of the loaded solvent is pumped through loaded solvent pumps and cooled in loaded solvent coolers before it is used in the H_2S absorber.

The loaded solvent routed for captured CO_2 isolation passes through three series of flash drums that operate at descending pressure. The high pressure drum recovers the H_2 by flashing the loaded solvent. The recovered H_2 is then compressed in H_2 recovery compressor and cooled through H_2 recovery coolers and returned to the CO_2 absorber syngas inlet. The medium pressure drum and low pressure drum recover the CO_2, while flashing the solvent out. The recovered CO_2 is then sent out the AGR unit to the CO_2 compression unit.

8.1.4. The SELEXOL Makeup Flow

SELEXOL makeup flow is provided from a constant solvent makeup inventory through the solvent makeup pump. Makeup flow is provided to the AGR unit through a connection to the inlet of the lean solvent pumps.

8.2. The Acid Gas Removal Unit Model

A high validity, non-linear model of the IGCC power plant including the AGR unit is available at AVESTAR center at WVU [113]. The model is implemented in a process modeling software called Dynsim®. Dynsim® is a process simulation environment [114], [115] designed for operator training and process design and research support. It features the ability to run a single process model across multiple machines up to 100 times faster than real time depending on the model complexity and computer configuration. Interested reader are referred to reference [111] for more detailed unit description and modeling.

An engine link developed by Schneider Electric was used to facilitate variable exchange between Matlab® and Dynsim® using the object linking and embedding for process control (OPC) data access protocol. The engine link utilizes a data mapping file, which assigns each desired Dynsim® point to an OPC data point and organizes them into OPC groups, based on user selection. Matlab® code must take into account the data

mapping file naming convention in order to access the values of the point. The Dynsim® - Matlab® engine link diagram is presented in Figure 8-2.

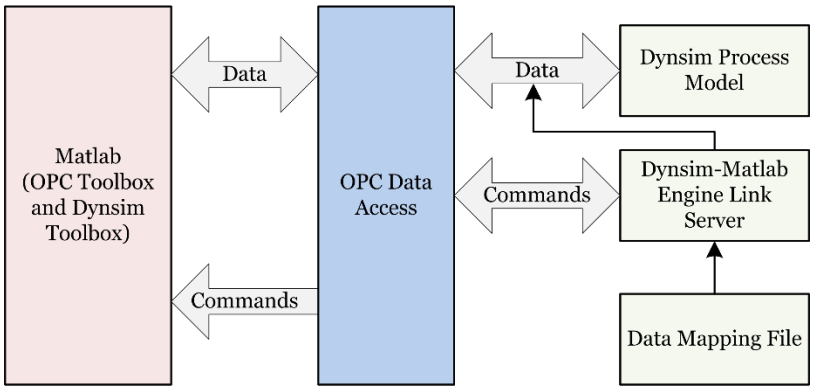

Figure 8-2. Dynsim®-Matlab® Engine Link Diagram

8.3. The Acid Gas Removal Unit Nominal Operations

The normal operation was defined using 10 operational constraints, as listed in Table 8-3. A sensitivity analysis was performed to determine all the controlled features that could affect the constraints. The analysis was performed by varying one controlled variabled at a time in heuristically pre-defined ranges and monitoring the constraints. The variable which had an impact on one or more constrains was recorded and used later to generate the self data.

Table 8-2. AGR Unit Normal Operational Constraints [116]

Operational Constraint	Value
Percentage CO_2 capture	95% of inlet CO_2
Percentage H_2S capture	99.95% of inlet H_2S
Solvent temperature at the outlet of the refrigeration coolers	4 °C
Solvent temperature at the outlet of the water coolers	21 °C
Maximum compressor power	+20% of nominal
Maximum heat exchanger duty	+50% of nominal
Maximum allowable solvent temperature	175 °C
Maximum allowable water content of solvent	6%
Minimum stripper pressure	276 kPa
Minimum Claus feed purity	25% H_2S

8.4. The Acid Gas Removal Unit Abnormal Conditions

The AGR unit could suffer several ACs. The distillation columns could accumulate solids deposits. The heat exchangers tubes could have leakage that results in contamination and hazardous situation. A list of such AC in the AGR unit considered in this study is presented in Table 8-3.

Table 8-3. AGR Unit ACs List

#	AC Name	Severity	#	AC Name	Severity
1	Reduction in the area of the 13th tray of the CO_2 absorber	15%	8	Leakage in the H_2 recovery compressor suction line	1%
		20%			2%
		25%			3%
2	Reduction in the area of the 15th tray of the CO_2 absorber	15%	9	Leakage in the H_2 recovery flash drum liquid phase	1%
		20%			2%
		25%			3%
3	Reduction in the area of the 23rd tray of the H_2S absorber	15%	10	Leakage in the H_2S acid gas knock-out drum liquid phase	1%
		20%			2%
		25%			3%
4	Reduction in the area of the 26th tray of the H_2S absorber	15%	11	Leakage in the CO_2 low pressure flash drum vapor phase	1%
		20%			2%
		25%			3%
5	Reduction in the area of the 4th tray of the H_2S concentrator	15%	12	Leakage in the CO_2 medium pressure flash drum vapor phase	1%
		20%			2%
		25%			3%
6	Reduction in the area of the 6th tray of the H_2S concentrator	15%	13	Reduction in the area of the 8th tray of the SELEXOL stripper	15%
		20%			20%
		25%			25%
7	Reduction in the heat transfer coefficient of lean/rich H.E	15%	14	Reduction in the area of the 11th tray of the SELEXOL stripper	15%
		20%			20%
		25%			25%

Reduction in heat transfer coefficient was simulated by reducing the heat transfer coefficient in the simulation by the percentage amount of the AC severity. Solids deposit on the trays of the distillation towers is simulated in Dynsim® by reducing the flow area of the affected tray by the percentage of the AC severity. Finally, leakages are simulated by modifying the affected flow line by splitting it into two streams - one continues to the original destination, while the other stream, representing the AC severity leakage, is routed to a sink.

Chapter 9. Results and Discussions

This chapter present the results of implementation of the proposed monitoring and control methodology for advanced power plant. The chapter is organized in three main sections. The first section demonstrates the immunity based ACDIC scheme as described in Chapter 5. The second section presents the use of the adaptive controllers described in Chapter 6 to augment different baseline controllers to ensure safer operation under AC. The final section demonstrates the use of evolutionary optimization algorithm described in Chapter 7 to optimize a PID parameters. These analyses were performed in Matlab® and Dynsim®. A series of Matlab® tools were developed to demonstrate the effectiveness of the proposed approach and allow future use and improvement. Those tools are presented in Chapter 10.

9.1. Demonstration of ACDIE Scheme

Features Selection: A total of 164 features for the entire 22 unit subsystems were selected to build the self of the AGR unit. The features included pressure, temperature, flow rate, and composition measurements across the unit. Those features were selected from an existing process and piping diagram of the unit [111].

Preliminary Self Data Preparation: The high fidelity Dynsim® model of the AGR was used for data collection. First, normal operation was defined using the 10 operational constraints, as listed in 8-2. A total of 729 tests, each lasting 4.5 hours were performed. In each test, all controlled features known to affect the constraints were varied in heuristically pre-defined ranges and the values of the selected features were recorded. The tests for which all constraints were met were used in building the self. The ranges of each feature in the normal test data were used to linearly normalize the feature values between 0 and 1.

HMS Strategy: Using the HMS strategy, the features were distributed in groups that completely defined one actual subsystem. Using this configuration also allows for AC identification without training, as described in section 5.5.1. Due to the computational limitations in Matlab®-Dynsim® engine caused by exchanging too many variables, only subsystems associated with one or more AC (as listed in Table 8-3) were considred. The list of the implemented subsystems, along with their features, are presented in Table 9-1.

Table 9-1. The AGR Selected Features

Subsystem	#	Tag	Feature Description
The H_2S Absorber	1	PT020	H_2S Absorber Sump Pressure
	2	FT010	Syngas Flow Rate from H_2S Absorber Feed Cooler
	3	FT001	Loaded Solvent Coolers to H_2S Absorber Reflux Flow Rate
	4	AT006A	Syngas CO_2 Composition to H2S Absorber
	5	AT006B	Syngas H_2S Composition to H2S Absorber
	6	PT006	Loaded Solvent Pressure from Loaded Solvent Cooler
	7	AT002A	Syngas CO_2 Composition to CO_2 Absorber
	8	AT002B	Syngas H_2S Composition to CO_2 Absorber
	9	FT003	Rich Solvent Flow Rate to Lean/Rich Heat Exchanger
	10	PDT002	H_2S Absorber Differential Pressure
The CO_2 Absorber	11	PT001	CO_2 Absorber Sump Pressure
	12	FT046	LP Flash Drum Solvent Flow Rate
	13	PT022	H_2S Absorber to CO_2 Absorber Pressure
	14	AT002A	Syngas CO_2 Composition from H_2S Absorber
	15	AT002B	Syngas H_2S Composition from H_2S Absorber
	16	FT020	CO_2 Reflux Flow from Lean Solvent Coolers
	17	PT003	Lean Solvent Pressure from Lean Solvent Cooler
	18	AT001A	Treated Syngas CO_2 Composition
	19	AT001B	Treated Syngas H_2S Composition
	20	PT021	CO2 Absorber Overhead Pressure
	21	PT027	Solvent Pressure from Solvent Chiller
The H_2S Concentrator	22	FT004	Solvent Flow Rate from H_2S Concentrator to H_2S Stripper
	23	AT004	Stripped Gas CO_2 Composition to Stripped Gas Cooler
	24	PDT004	H_2S Concentrator Differential Pressure
	25	ST16P	Pressure to the Stripped Gas Coolers
	26	ST16W	Flow Rate to the Stripped Gas Coolers
	27	ST15P	Pressure from Lean/Rich Solvent Heat Exchanger
	28	ST15W	Flow Rate from Lean/Rich Solvent Heat Exchanger
	29	ST21P	Reheated Syngas Pressure
	30	ST21W	Reheated Syngas Flow
	31	ST18P	Pressure to H2S Absorber
	32	ST18W	Flow Rate to H2S Absorber
The Lean/Rich Solvent Heat Exchanger	33	FT003	Rich Solvent Flow Rate from H_2S Absorber
	34	PT040	Rich Solvent Pressure to H_2S Concentrator
	35	TT040	Rich Solvent Temperature H_2S Concentrator
	36	TT035	Lean Solvent Temperature to Lean Solvent Coolers
	37	PT002	Pressure to Lean Solvent Cooler (AA)
	38	TT002	Temperature to Lean Solvent Cooler(AA)
	39	TT004	Temperature to Lean Solvent Cooler(AB)
	40	PT004	Pressure to Lean Solvent Cooler (AB)
	41	FT012	Lean Solvent Flow Rate to Lean Solvent Coolers

Table 9-1. The AGR Selected Features (cont.)

Subsystem	#	Tag	Feature Description
The SELEXOL Stripper	42	PT030	H_2S Stripper Sump Pressure
	43	FT005	H_2S Stripper Reflux Flow Rate
	44	FT011	Acid Gas Vapor Flow Rate to the Acid Gas Coolers
	45	PDT003	H_2S Stripper Differential Pressure
	46	ST32P	Pressure from the Reboiler
	47	ST32W	Flow Rate from the Reboiler
	48	ST04W	Pressure to the Acid Gas Coolers
	49	ST51P	Pressure from the Reflux Pumps
	50	ST51W	Flow Rate from the Reflux Pumps
	51	ST64P	Pressure from the H_2S Concentrator
	52	ST64W	Flow Rate from the H_2S Concentrator
The H_2 Recovery System	53	ST015	H2 Recovery Compressor Speed
	54	PT015	H_2 Recovery Compressor Inlet Pressure
	55	PT016	H_2 Recovery Compressor Outlet Pressure
	56	PT010	H_2 Recovery Pressure to H_2 Recovery Cooler (AA)
	57	PT012	H_2 Recovery Pressure to H2 Recovery Cooler (BA)
	58	D1001P	H_2 Recovery Drum Pressure
	59	D1001T	H_2 Recovery Drum Temperature
	60	PT011	H_2 Recovery Pressure from H_2 Recovery Cooler (AB)
	61	PT013	H_2 Recovery Pressure from H_2 Recovery Cooler (BB)
The Acid Gas Knock Out Drum	62	PT060	Acid Gas Knock Out Drum Pressure
	63	FT013	Makeup Water Flow Rate to Acid Gas Knock Out Drum
	64	TT043	Acid Gas Temperature Entering from the Acid Gas Coolers
	65	AT003	Acid Gas Composition to Claus Plant
	66	FT014	Reflux Bleed Flow Rate from Acid Gas Knock Out Drum
The Medium Pressure Flash Drum	67	D1003P	Medium Pressure Flash Drum Pressure
	68	D1003T	Medium Pressure Flash Drum Temperature
	69	PT033	Pressure of Exiting Gas to Acid Gas Knock Out Drum
	70	TT023	Temperature of Exiting Gas to Acid Gas Knock Out Drum
The Low Pressure Flash Drum	71	D1004P	Low Pressure Flash Drum Pressure
	72	D1004T	Low Pressure Flash Drum Temperature
	73	PT034	Pressure of Exiting Gas to CO_2 Compressor
	74	TT024	Temperature of Exiting Gas to CO_2 Compressor
	75	PT026	Solvent Pressure from LP Flash Drum to Solvent Chillers (AA)
	76	PT028	Solvent Pressure from LP Flash Drum to Solvent Chillers (BA)

Self Generation Using the PUA: A universe of hyper cubes with size equal to 0.025 and dimensionality equal to each subsystem feature number was generated. The data were compared against the universe grid producing the self, as described in Chapter 4. Table 9-2 presents the subsystem name, the number of features, and the number of resulted partition clusters.

Table 9-2. The Targeted Subsystems and Resulted Selves Dimensions

Subsystem	Number of Features	Number of Self Partitions
The H_2S Absorber	10	21248
The CO_2 Absorber	11	67381
The H_2S Concentrator	11	55575
Lean/Rich Solvent Heat Exchanger	9	420795
The SELEXOL Stripper	11	1217921
The H_2 Recovery System	9	18100
The Acid Gas Knock Out Drum	5	390281
The Medium Pressure Flash Drum	4	10735
The Low Pressure Flash Drum	6	93360

The Artificial DC Algorithm: The Artificial DC algorithm described in section 5.2 was implemented in Matlab®. The following parameters were used: the number of immature DCs $N_{DC} = 100$, the time window size $T = 15$ samples, the selection rate $\sigma = 0.65$, all initial DCs life $\mathcal{L} = 15$ samples, and all initial DCs migration threshold $\mathcal{M} = 5$ samples. The interleukin 10 and 12 update functions used are described in Eq. (5-23) and Eq.(5-27) respectively.

9.1.1. AC Detection Performance

The performance of the detection scheme is evaluated using three metrics, the false alarm rate (FA), detection time (DT), and detection rate (DR) as described in section 5.8.

Three different validation tests under nominal conditions were obtained in a similar manner as for the data used for building the self. These validation tests were only used to calculate the false alarm rate. 0% FA was obtained from all the validation tests.

The ACs summarized in Table 8-3 were implemented in Dynsim®. Each AC were run three time each starting from a different initial conditions for a total of 126 tests. The AC occurrence time was randomly selected and each test lasted for 20 minutes simulation time. It is worth mentioning that process model sample time was 0.25 seconds, while the ACDIE algorithms sample time was 5 seconds. The reason for this setting is to speed up the simulation and reduce the Dynsim®-Matlab® engine computational overhead. Table 9-3 summarizes the average detection outcomes for all targeted AC.

Table 9-3. AC Detection Performance

#	AC Name	Severity	DT (sec)	DR (%)
1	Reduction in the area of the 13th tray of the CO_2 absorber	15%	220.75	100
		20%	229.08	100
		25%	134.33	100
2	Reduction in the area of the 15th tray of the CO_2 absorber	15%	225.75	100
		20%	215.75	100
		25%	127.42	100
3	Reduction in the area of the 23rd tray of the H_2S absorber	15%	57.42	100
		20%	60.75	100
		25%	57.42	100
4	Reduction in the area of the 26th tray of the H_2S absorber	15%	59.08	100
		20%	57.42	100
		25%	57.42	100
5	Reduction in the area of the 4th tray of the H_2S concentrator	15%	57.75	100
		20%	54.08	100
		25%	52.42	100
6	Reduction in the area of the 6th tray of the H_2S concentrator	15%	55.75	100
		20%	50.75	100
		25%	50.75	100
7	Reduction in the heat transfer coefficient of lean/rich H.E	15%	65.75	100
		20%	62.50	100
		25%	59.08	100
8	Leakage in the H_2 recovery compressor suction line	1%	85.75	100
		2%	80.75	100
		3%	79.08	100
9	Leakage in the H_2 recovery flash drum liquid phase	1%	65.75	100
		2%	62.42	100
		3%	65.75	100
10	Leakage in the H_2S acid gas knock-out drum liquid phase	1%	670.75	100
		2%	484.75	100
		3%	402.5	100
11	Leakage in the CO_2 low pressure flash drum vapor phase	1%	180.75	100
		2%	190.75	100
		3%	130.75	100
12	Leakage in the CO_2 medium pressure flash drum vapor phase	1%	110.75	100
		2%	89.08	100
		3%	70.75	100
13	Reduction in the area of the 8th tray of the SELEXOL stripper	15%	90.75	87.09
		20%	55.75	100
		25%	57.42	100
14	Reduction in the area of the 11th tray of the SELEXOL stripper	15%	60.75	100
		20%	57.42	100
		25%	57.42	100

Detection performance is excellent with detection time ranging between 50.75 and 670 seconds. The average of the detection time is 122 seconds and may be rated as excellent, considering the time scale of the targeted system. The larger *DT* recorded for AC9 is attributed to the slow reaction of selected features and may be improved by including additional features. The monitoring scheme achieves 100% *DR* for almost all cases. It should be noted that for low severity of AC14, the lower *DR* is due to a minimal overlap between self/non-self. Addition of more features or selection of higher partition resolution is expected to improve this evaluation parameter in this case. In summary, the proposed DC algorithm for AC detection provided excellent false alarms, detection rates, and detection times for all the targeted ACs.

9.1.2. AC Identification Performance

The result for all AC identification are presented in Table 9-4.

Table 9-4. AC Identification Performance

#	AC Name	Severity	IR	#	AC Name	Severity	IR
1	Reduction in the area of the 13th tray of the CO_2 absorber	15%	100	8	Leakage in the H_2 recovery compressor suction line	1%	100
		20%	100			2%	100
		25%	100			3%	100
2	Reduction in the area of the 15th tray of the CO_2 absorber	15%	100	9	Leakage in the H_2 recovery flash drum liquid phase	1%	100
		20%	100			2%	100
		25%	100			3%	94.41
3	Reduction in the area of the 23rd tray of the H_2S absorber	15%	100	10	Leakage in the H_2S acid gas knock-out drum liquid phase	1%	100
		20%	100			2%	100
		25%	100			3%	100
4	Reduction in the area of the 26th tray of the H_2S absorber	15%	100	11	Leakage in the CO_2 low pressure flash drum vapor phase	1%	100
		20%	100			2%	100
		25%	100			3%	99.4
5	Reduction in the area of the 4th tray of the H_2S concentrator	15%	94.0	12	Leakage in the CO_2 medium pressure flash drum vapor phase	1%	100
		20%	92.85			2%	100
		25%	92.27			3%	100
6	Reduction in the area of the 6th tray of the H_2S concentrator	15%	93.42	13	Reduction in the area of the 8th tray of the SELEXOL stripper	15%	100
		20%	92.14			20%	100
		25%	91.70			25%	100
7	Reduction in the heat transfer coefficient of lean/rich H.E	15%	100	14	Reduction in the area of the 11th tray of the SELEXOL stripper	15%	100
		20%	100			20%	100
		25%	100			25%	100

The goal of the AC identification is to isolate the most affected subsystem. The subsystem pattern approach described in section 5.5.1 was used for the AC identification purpose. The subsystem features are grouped as shown in Table 8-1. The number of self

projections N_p is equal to the number of targeted subsystems, that is 9. Once a self projection is triggered, the algorithm identifies the corresponding subsystem. In some instances, more than one self projections are triggered; therefore, previous time sample identification outcomes were used to solve the conflict.

The monitoring scheme achieves excellent IR. In only 8 cases IR is less than 100%, but not less than 91%. It proves that the training-free subsystem pattern approach proposed in this research is an excellent approach to address the AC identification problem. It eliminates the need for training of the pattern matching algorithm and the computational issues of the on-line matching process, while still providing excellent IR outcomes.

9.1.3. AC Evaluation Performance

The partition tracking approach described in section 5.6.1 was used to address the AC evaluation problem. The naïve Bayes classifier was trained using a set of training AC tests to define the reference partition tracking patterns for each failure type and severity. Figure 9-1 to Figure 9-10 present the reference partition tracking patterns for selected subsystems AC types and severities. Note that the AC evaluation approach scheme was tested for the scenario when only one subsystem is under a single AC with specific type and severity. Table -9-5 shows the AC TER and SER rates for the all targeted AGR ACs. The TER and SER are calculated as presented in Eq.(5-68) and Eq. (5-69) respectively.

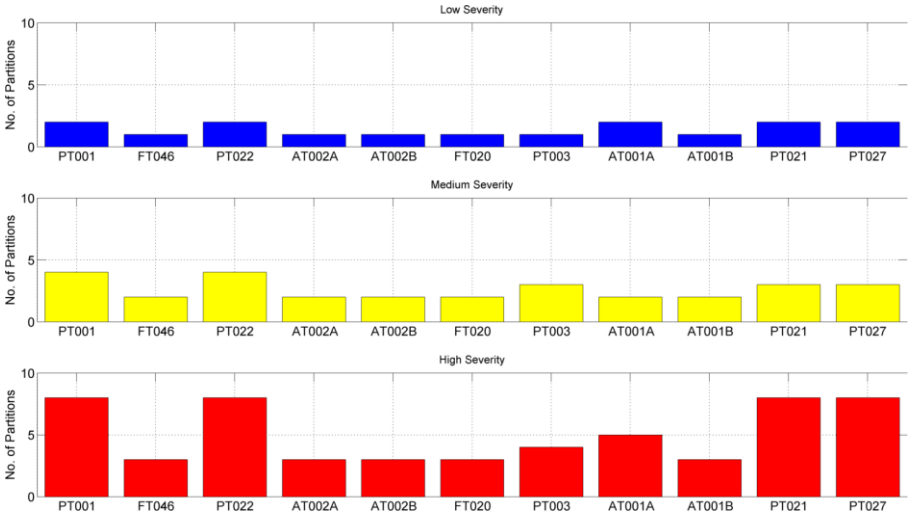

Figure 9-1. Reference Partition Tracking Pattern for AC1 and AC2

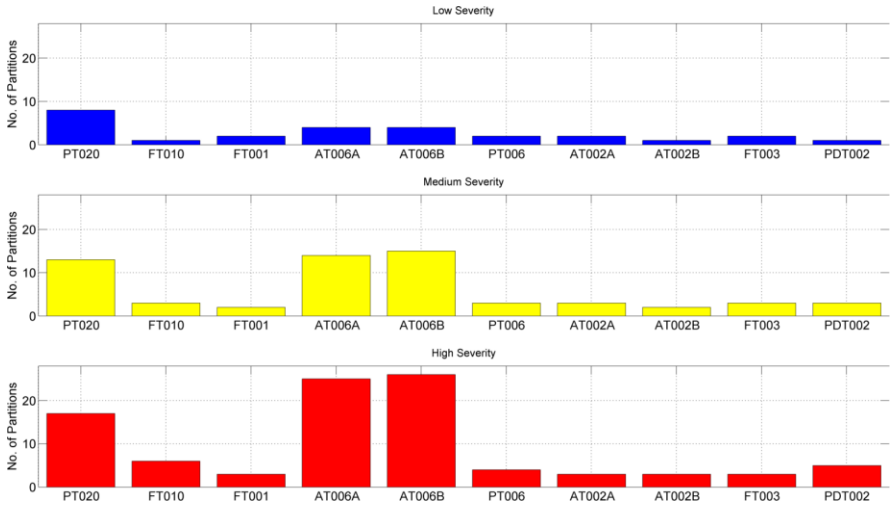

Figure 9-2. Reference Partition Tracking Pattern for AC3 and AC4

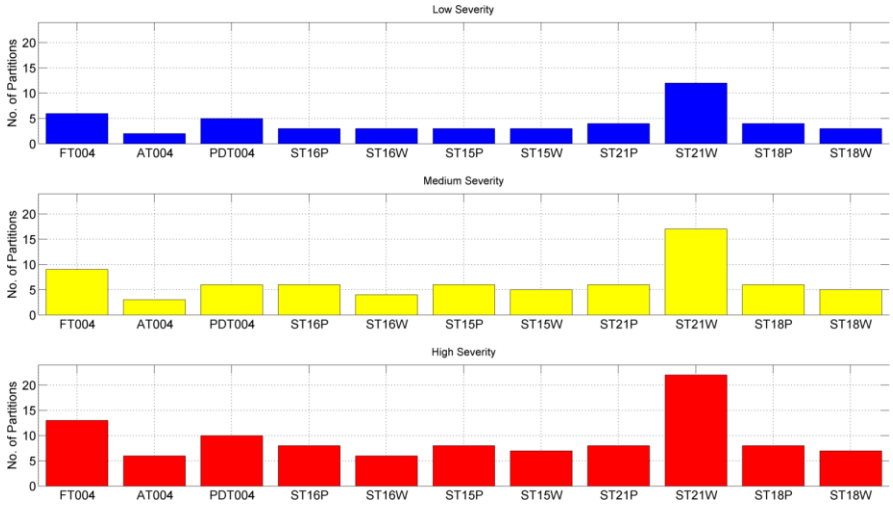

Figure 9-3. Reference Partition Tracking Pattern for AC5 and AC6

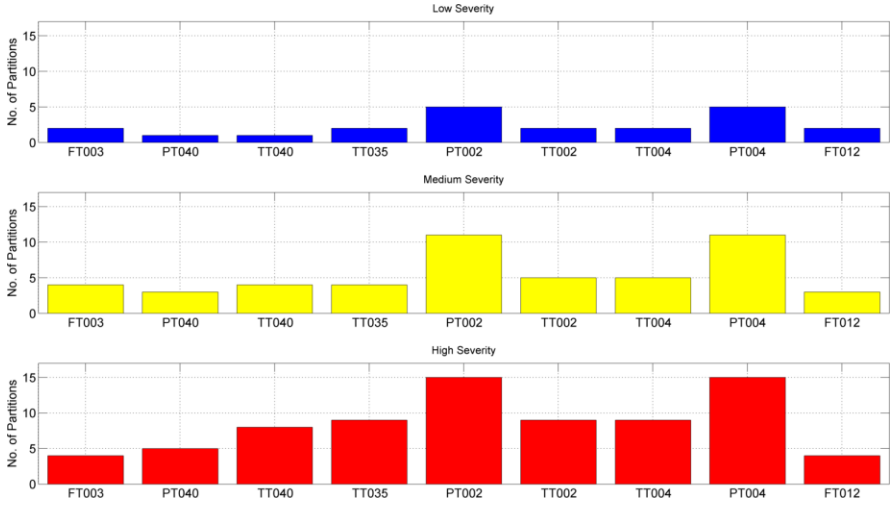

Figure 9-4. Reference Partition Tracking Pattern for AC7

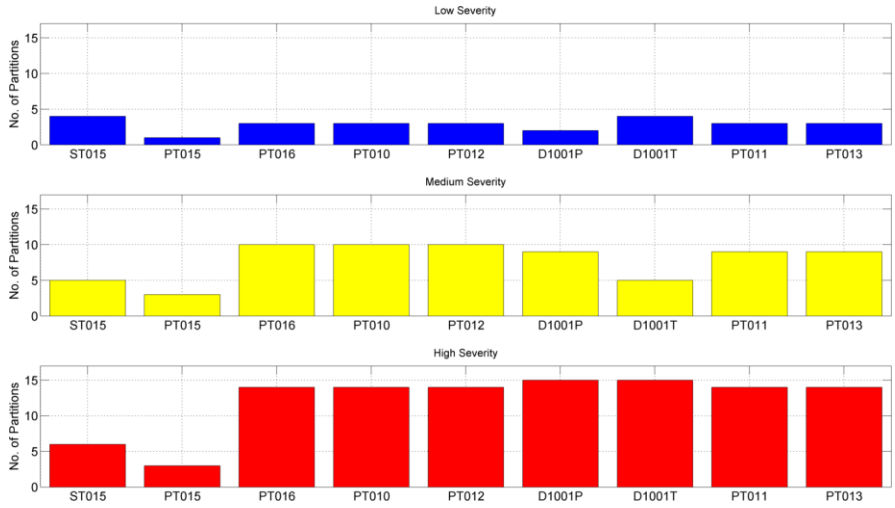

Figure 9-5. Reference Partition Tracking Pattern for AC8

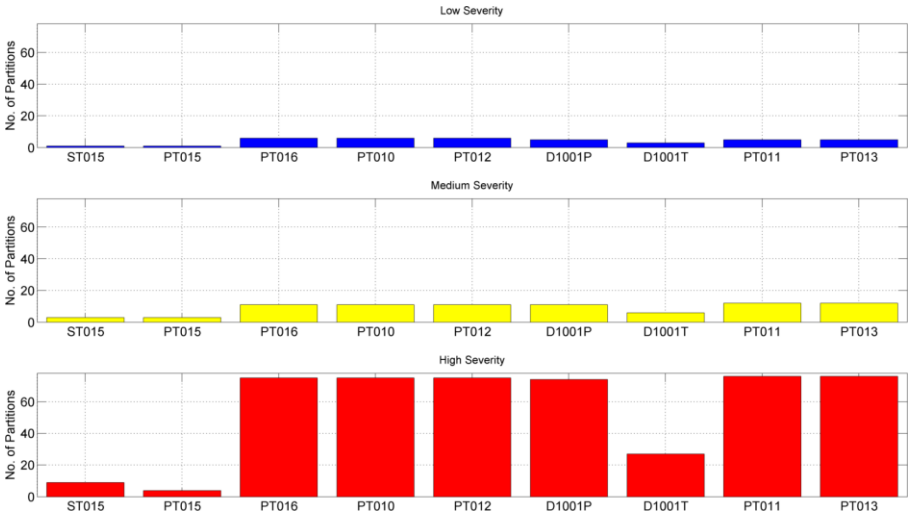

Figure 9-6. Reference Partition Tracking Pattern for AC9

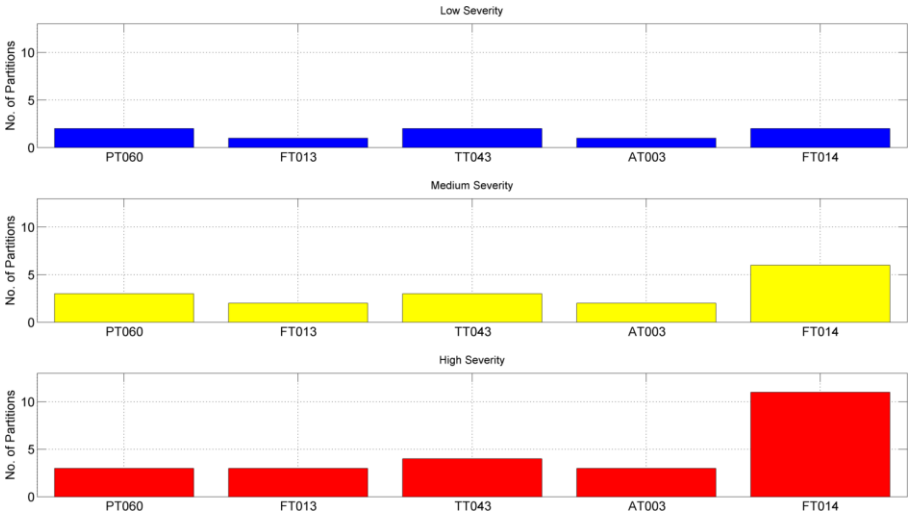

Figure 9-7. Reference Partition Tracking Pattern for AC10

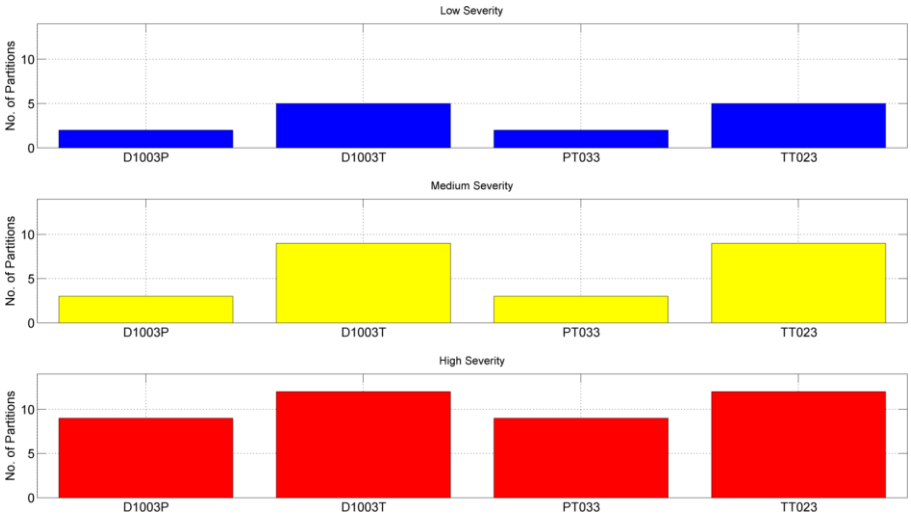

Figure 9-8. Reference Partition Tracking Pattern for AC11

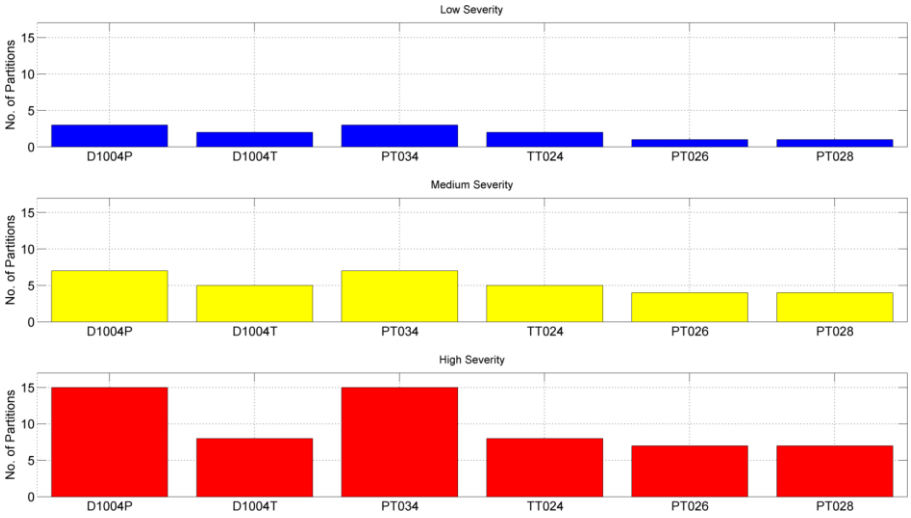

Figure 9-9. Reference Partition Tracking Pattern for AC12

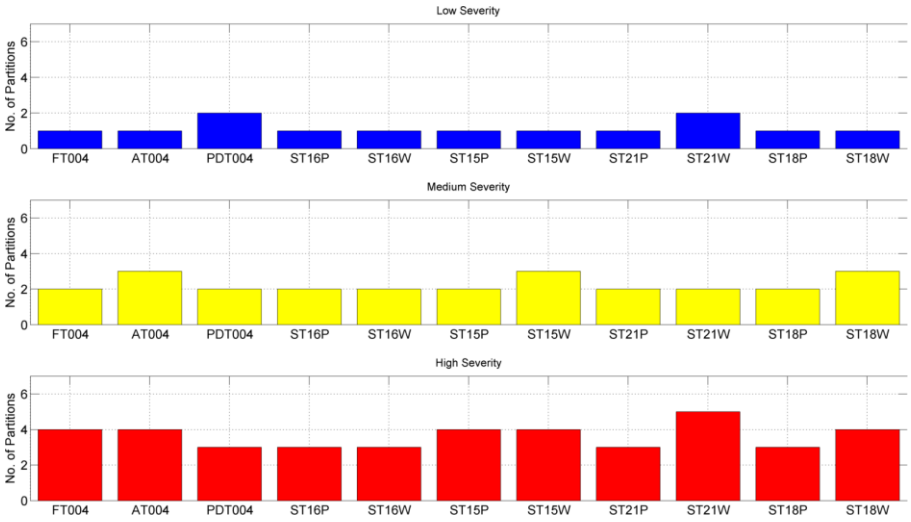

Figure 9-10. Reference Partition Tracking Pattern for AC13 and AC14

Table -9-5. AC Evaluation Performance

#	AC Name	Severity	TER (%)	SER (%)
1	Reduction in the area of the 13th tray of the CO_2 absorber	15%	100	100
		20%	100	78.28
		25%	100	95.13
2	Reduction in the area of the 15th tray of the CO_2 absorber	15%	100	89.00
		20%	100	81.29
		25%	100	94.85
3	Reduction in the area of the 23rd tray of the H_2S absorber	15%	100	100
		20%	100	83.70
		25%	100	82.28
4	Reduction in the area of the 26th tray of the H_2S absorber	15%	100	100.00
		20%	100	80.81
		25%	100	82.27
5	Reduction in the area of the 4th tray of the H_2S concentrator	15%	100	100
		20%	100	96.86
		25%	100	98.58
6	Reduction in the area of the 6th tray of the H_2S concentrator	15%	100	86.85
		20%	100	78.67
		25%	100	100
	Reduction in the heat transfer coefficient of lean/rich H.E	15%	100	100
		20%	100	94.84
		25%	100	83.72
8	Leakage in the H_2 recovery compressor suction line	1%	98.65	100
		2%	98.21	100
		3%	97.76	89.01
9	Leakage in the H_2 recovery flash drum liquid phase	1%	100	92.77
		2%	100	87.79
		3%	100	56.34
10	Leakage in the H_2S acid gas knock-out drum liquid phase	1%	100	91.43
		2%	100	40.05
		3%	100	54.62
11	Leakage in the CO_2 low pressure flash drum vapor phase	1%	100	100
		2%	100	78.61
		3%	100	72.14
12	Leakage in the CO_2 medium pressure flash drum vapor phase	1%	100	88.94
		2%	100	99.55
		3%	100	97.47
13	Reduction in the area of the 8th tray of the SELEXOL stripper	15%	100	100
		20%	100	98.25
		25%	100	44.93
14	Reduction in the area of the 11th tray of the SELEXOL stripper	15%	100	83.70
		20%	100	52.85
		25%	100	96.49

The obtained AC evaluation results show the capability of the proposed partition pattern tracking approach in isolating the AC type with very high success rates. The performance of the monitoring scheme in terms of AC severity evaluation may appear lower due to a typical underestimation of higher severity levels. It should be noted that the average SER for low severity ACs is 96%. The performance for more severe ACs declines because the feature points migrate first from self regions into non-self region that correspond to low severity regions, hence the initial severity underestimation. This feature point migration under ACs reflects the accumulation over time of AC effects on the system.

9.2. Demonstration of Adaptive Control Mechanisms

The adaptive mechanisms presented in Chapter 6 are tested using different baseline controllers under different configurations. The next sections describe the baseline controllers and the configuration used to test adaptive mechanism performance.

9.2.1. Demonstration Using Linearized Model

The CO_2 absorption process unit of the IGCC-AGR process is selected as a subsystem for the primary case study. This unit is of most importance because it facilitates the capture of CO_2 generated in the coal gasification unit of the IGCC process. A 2-input-2-output sub-system is selected from the CO_2 absorption process unit as a case study. It is observed that the flow rate and the temperature of the incoming recycled solvent stream greatly affect the CO_2 capture from the CO_2 absorber process unit. Therefore, the percentage CO_2 present in the outlet stream from the CO_2 absorber unit and the temperature of the recycled solvent stream feeding into the CO_2 absorber unit are selected as desired output variables, while the flow rate of the recycled solvent stream coming in and the flow rate of the refrigerant required to cool down this stream are defined as the input or manipulated variables. Using data collected from the Dynsim® model of the AGR, a linearized 2-inputs/2-outputs/4-states model of this process was obtained by Mirlekar et al. [117], [118]. The state space representation of the linearized models are considered to be the controlled plant (see Figure 9-11). Two main verification scenarios (VS) have been considered:

VS#1: Commanded trajectories defined as constant setpoints in terms of the percentage CO_2 present in the outgoing stream from the absorber and the temperature of the recycled solvent stream going into the absorber were considered. The commanded

trajectories were tracked using: PID controller, PID augmented with ANN-based adaptive mechanism, and PID augmented with immunity-based mechanism. Note that two versions of the ANN-augmented control laws were considered. One uses as ANN inputs all the states of the plant, while the other uses outputs of the plant instead. The general block diagram of the testing configuration is illustrated in Figure 9-11. Testing under both normal and abnormal conditions were performed. The integral of the absolute values of the tracking error was used as a performance metric. U notation was used to indicate unstable responses.

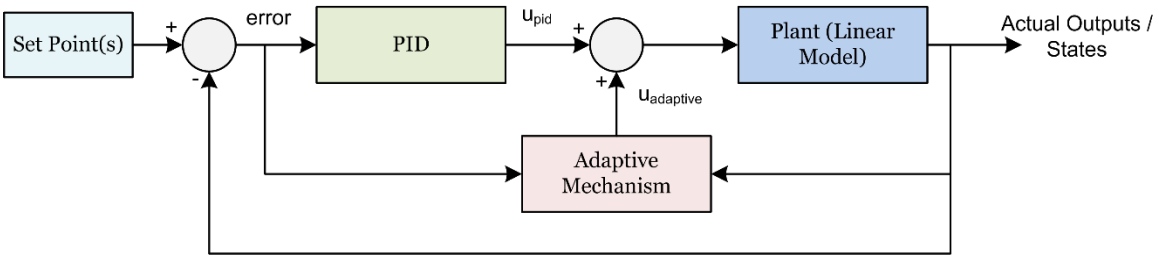

Figure 9-11. Adaptive Control Testing Using PID Controller and a Linearized Model

The results are listed in Table 9-6. The baseline controller (i.e. PID) gains were optimized for nominal operation to ensure minimum tracking errors using the linearized model of the plant. Therefore, the performance of the baseline controller under normal condition is very good, as expected. The PID controller was then augmented with the adaptive elements and tested at normal conditions. The adaptive elements are expected to enhance or at least maintain the baseline controller performance under normal conditions. The results show that the combination of the baseline controller with all of the adaptive elements provided very good performance under nominal conditions.

The performance of the stand-alone baseline controller was then assessed under the occurrence of abnormal conditions. The abnormal conditions are simulated by the alteration of the non-zero elements of A and B matrices in the state space representation of the linearized model. The results confirm the expectation that the baseline controller features limited robustness and its performance degrades under abnormal conditions up to the point of losing stability. However, the adaptive augmentation, in all cases, increases robustness resulting in maintaining stability and improving tracking performance. It is worth mentioning that the immunity ACDIE may allow for a systematic selection for the best adaptive controller under different AC operations.

Table 9-6. Tracking Performance of PID and Different Adaptive Controllers

System Condition		System Modes (*10^{-3})	Damp Ratio	Nat. Freq. (*10^{-3})	PID	PID + ANN States	PID + ANN Outputs	PID + AIS
Nominal		{-0.7 ± i, -13.6, -0.6}	{0.59, 1 , 1 }	{1.3, 13.6, 0.6}	143.68	33.08	84.45	15.80
Actuator Failure	B(1,1)*2	=	=	=	79.45	25.04	47.65	9.78
	B(1,1)/2	=	=	=	143.68	33.08	84.45	15.80
	B(2,1)*2	=	=	=	144.06	33.18	84.90	15.81
	B(2,1)/2	=	=	=	143.68	33.08	84.45	15.80
Plant Failure	A(1,1)*1.5	{-17.9, 7.7 -13.6,-0.6}	{1, -1, 1, -1}	{17.9, 7.7, 13.6, 0.6}	U	33.76	650.84	674.79
	A(1,1)/1.5	{2.2±9.4i, -13.6, -0.6}	{-0.22 1, 1}	{9.7, 13.6, 0.6}	U	29.53	188.15	563.30
	A(1,2)*2	{-0.7±16.7i, -13.6, -0.6}	{0.05, 1, 1}	{16.7, 13.6, 0.6}	U	20.64	186.23	560.66
	A(1,2)/1.25	{-8.1, 6.6, -13.6, -0.6}	{1, -1, 1, 1}	{8.1, 6.6, 13.6, 0.6}	U	32.36	353.07	463.63
	A(2,1)*1.5	{-0.7±11.8i, -13.6, -0.6}	{0.06, 1, 1}	{11.9, 13.6, 0.6}	U	20.32	187.33	563.80
	A(2,1)/1.5	{-10.3, 8.8, -13.6, -0.6}	{1, -1, 1, 1}	{10.3, 8.8, 13.6, 0.6}	U	36.30	875.67	674.70
	A(2,2)*2	{-7.1, 9.6, -13.6, -0.6}	{1, -1, 1, 1}	{7.1, 9.6, 13.6, 0.6}	U	38.90	1.32e+3	664.55
	A(2,2)/3	{-6.0±12.2i, -13.6, -0.6}	{0.44, 1, 1}	{13.6, 13.6, 0.6}	U	15.18	157.79	473.10
	A(3,3)*1.5	{-0.7±i, -30.4, 3.9}	{0.6, 1, -1}	{1.3, 30.4, 3.9}	143.66	33.07	84.45	15.81
	A(3,3)/1.5	{-0.7±i, -3.0±9.2i}	{0.59, 0.31}	{1.3, 9.7}	143.71	33.08	84.46	15.81
	A(3,4)*3	{-0.7±i, -7.1±22.0i}	{0.60, 0.31}	{1.3, 23.2}	143.76	33.08	84.52	15.73
	A(3,4)/1.1	{-0.7±i, -15.2, 1}	{0.60, 1, -1}	{1.3, 15.2, 1}	U	33.15	85.95	17.38
	A(4,3)*1.5	{-0.7±i, -7.1±9.5i}	{0.60, 0.60}	{1.3, 11.8}	143.70	33.08	84.45	15.81
	A(4,3)/1.5	{-0.7±i, -18.5, 4.3}	{0.60, 1, -1}	{1.3, 18.5, 4.3}	143.67	33.08	84.45	15.80
	A(4,4)*1.1	{-0.7±i, -14.4, 1.2}	{0.60, 1, -1}	{1.3, 14.4, 1.2}	U	33.48	90.58	27.62
	A(4,4)/1.5	{-0.7±i, -8.8±3.9i}	{0.60, 0.91}	{1.3, 9.7}	143.71	33.07	84.46	15.76

VS#2: Optimal commanded trajectories were obtained defined as variable setpoints in terms of the flow rate of the recycled solvent stream into the absorber and the flow rate of the refrigerant in the heat exchanger using a multi-agent-based algorithm that combines the ants' rule of pursuit idea with optimal control concepts for the calculation of optimal trajectories of individual agents [117]. These optimal trajectories were tested open loop with no adaptive augmentation (Figure 9-12), with ANN-based augmentation (Figure 9-13 and Figure 9-14), and immunity-based augmentation (Figure 9-15). Note that two versions of the ANN-augmented control laws were considered again: using all the states of the plant as ANN inputs (Figure 9-13) and using outputs of the plant instead (Figure 9-14). Testing under both normal and abnormal conditions were performed.

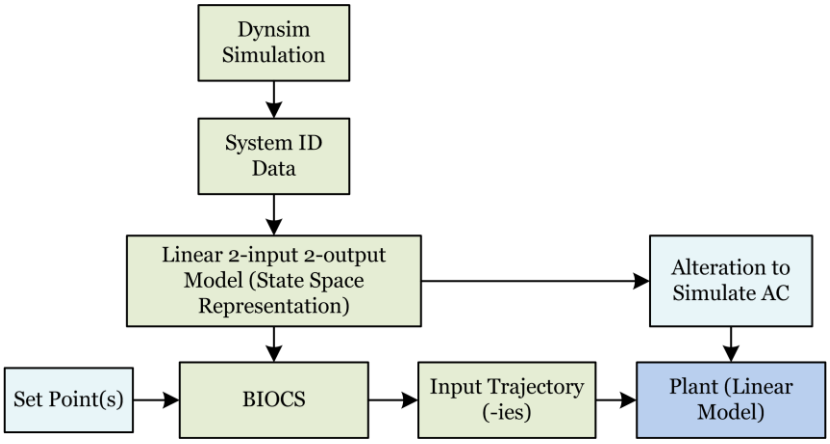

Figure 9-12. Optimal Trajectory Test with No Adaptive Augmentation

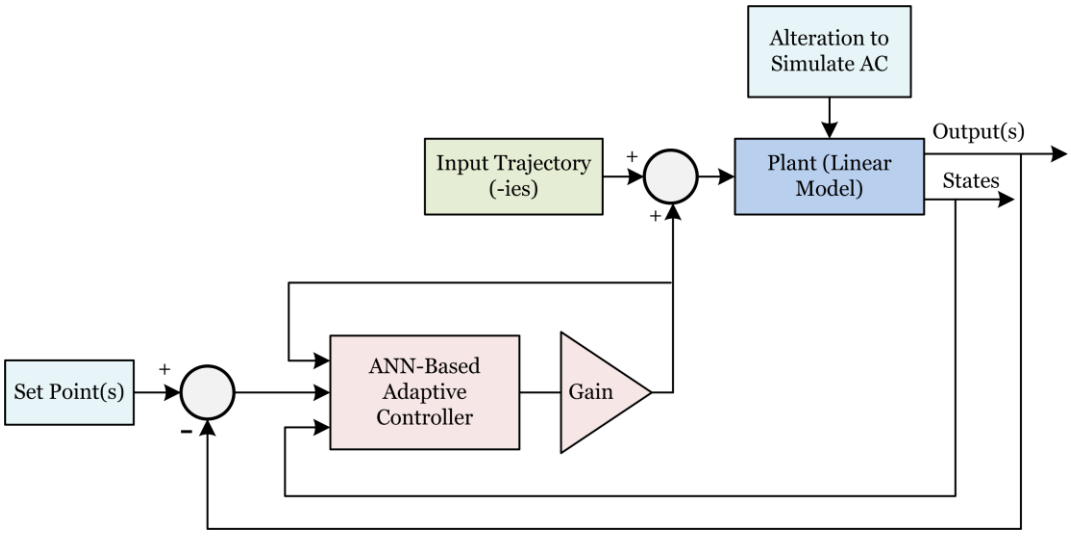

Figure 9-13. Optimal Trajectory Test with ANN-Plant-State-based Augmentation

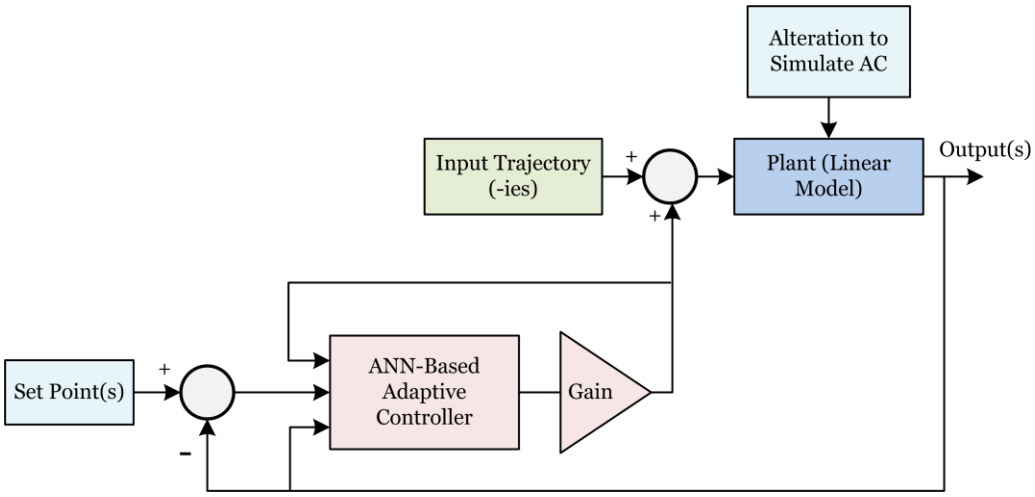

Figure 9-14. Optimal Trajectory Test with ANN-Plant Output-based Augmentation

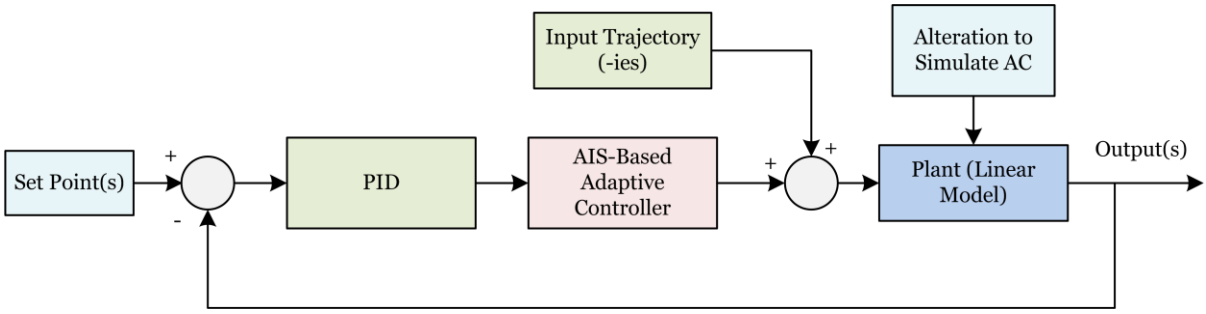

Figure 9-15. Optimal Trajectory Test with Immunity-based Augmentation

The results for VS#2 are listed in Table 9-7. The integral of the absolute values of the tracking error was used as a performance metric. U notation was used to indicate unstable responses. The baseline controller now consists of the trajectory generated through the biomimetic approach. Since this commanded trajectory was optimized under nominal conditions using the linearized plant model, the performance is very good at nominal conditions, as expected. All adaptive mechanisms typically maintain or slightly improve this performance under nominal conditions. When abnormal conditions are considered (simulated in a similar manner as for VS#1), the performance of the baseline controller output degrades. The adaptive augmentation, in all cases considered, is capable of increasing system robustness and maintains system stability with good tracking performance.

Table 9-7. Tracking Performance of Biomimetic and Different Adaptive Controllers

System Condition		System Modes ($*10^{-3}$)	Damp Ratio	Nat. Freq. ($*10^{-3}$)	Traj	Traj + ANN States	Traj + ANN Outputs	Traj + AIS
Nominal		{-0.7 ± i, -13.6, -0.6}	{0.59, 1 , 1 }	{1.3, 13.6, 0.6}	198.58	44.37	446.32	10.39
Actuator Failure	B(1,1)*2	=	=	=	$1.0*10^4$	42.07	658.86	11.30
	B(1,1)/2	=	=	=	$3.4*10^3$	50.24	312.58	18.19
	B(2,1)*2	=	=	=	$1.9*10^3$	44.70	484.37	10.40
	B(2,1)/2	=	=	=	$1.9*10^3$	43.84	413.18	10.36
Plant Failure	A(1,1)*1.5	{-17.9, 7.7 -13.6,-0.6}	{1, -1, 1, -1}	{17.9, 7.7, 13.6, 0.6}	U	39.60	146.01	536.83
	A(1,1)/1.5	{2.2±9.4i, -13.6, -0.6}	{-0.22 1, 1}	{9.7, 13.6, 0.6}	U	41.44	169.86	410.70
	A(1,2)*2	{-0.7±16.7i, -13.6, -0.6}	{0.05, 1, 1}	{16.7, 13.6, 0.6}	$7.7*10^3$	29.30	100.25	410.60
	A(1,2)/1.25	{-8.1, 6.6, -13.6, -0.6}	{1, -1, 1, 1}	{8.1, 6.6, 13.6, 0.6}	U	39.14	139.74	368.86
	A(2,1)*1.5	{-0.7±11.8i, -13.6, -0.6}	{0.06, 1, 1}	{11.9, 13.6, 0.6}	$7.7*10^3$	27.96	77.41	412.26
	A(2,1)/1.5	{-10.3, 8.8, -13.6, -0.6}	{1, -1, 1, 1}	{10.3, 8.8, 13.6, 0.6}	U	43.05	197.54	537.81
	A(2,2)*2	{-7.1, 9.6, -13.6, -0.6}	{1, -1, 1, 1}	{7.1, 9.6, 13.6, 0.6}	U	96.58	830.90	$5.6*10^3$
	A(2,2)/3	{-6.0±12.2i, -13.6, -0.6}	{0.44, 1, 1}	{13.6, 13.6, 0.6}	$7.7*10^3$	21.85	48.74	346.87
	A(3,3)*1.5	{-0.7±i, -30.4, 3.9}	{0.6, 1, -1}	{1.3, 30.4, 3.9}	$1.9*10^3$	44.37	444.74	10.39
	A(3,3)/1.5	{-0.7±i, -3.0±9.2i}	{0.59, 0.31}	{1.3, 9.7}	$1.9*10^3$	44.39	448.24	10.39
	A(3,4)*3	{-0.7±i, -7.1±22.0i}	{0.60, 0.31}	{1.3, 23.2}	$1.9*10^3$	44.35	446.75	10.38
	A(3,4)/1.1	{-0.7±i, -15.2, 1}	{0.60, 1, -1}	{1.3, 15.2, 1}	$1.9*10^3$	44.38	447.27	10.62
	A(4,3)*1.5	{-0.7±i, -7.1±9.5i}	{0.60, 0.60}	{1.3, 11.8}	$1.9*10^3$	44.39	448.65	10.39
	A(4,3)/1.5	{-0.7±i, -18.5, 4.3}	{0.60, 1, -1}	{1.3, 18.5, 4.3}	$1.9*10^3$	44.37	444.72	10.39
	A(4,4)*1.1	{-0.7±i, -14.4, 1.2}	{0.60, 1, -1}	{1.3, 14.4, 1.2}	$1.9*10^3$	44.39	447.47	10.78
	A(4,4)/1.5	{-0.7±i, -8.8±3.9i}	{0.60, 0.91}	{1.3, 9.7}	$1.9*10^3$	44.36	446.03	10.36

9.2.2. Demonstration Using Non-Linear Model

In this demonstration, the non-linear model of the subsystem described in section 9.2.1 was used. The baseline controller consists of a biologically inspired optimal control strategy (BIOCS) [117]. The BIOCS controller considered here mimics the ants' rule of pursuit in combination with optimal and agent based control concepts. The BIOCS is implemented to solve an optimal control problem associated with a multiple input multiple output system. The optimal control problem involves minimizing an objective function, while satisfying all constraint functions. In this case, the objective function consists of minimizing the error between multiple outputs and their desired setpoints.

The commanded trajectories considered were defined as constant setpoints in terms of the percentage CO_2 present in the outgoing stream from the absorber and the temperature of the recycled solvent stream going into the absorber were considered. The commanded trajectories were tracked using: BIOCS alone and the BIOCS augmented with ANN-based adaptive mechanism, as presented in Figure 9-16 and Figure 9-17.

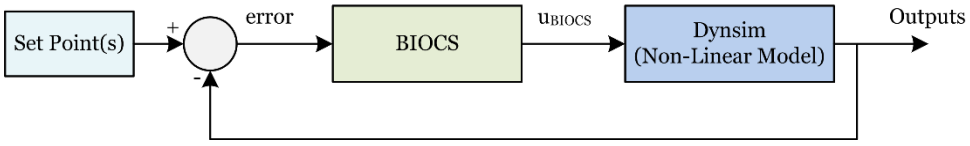

Figure 9-16. Standalone BIOCS Configuration

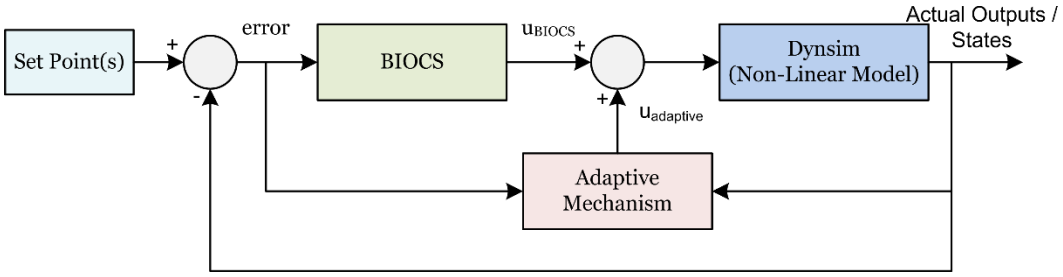

Figure 9-17. BIOCS and Adaptive Control Configuration

The objective of the BIOCS was to simultaneously track the CO_2 percentage in the outgoing stream and temperature of the recycled solvent. The BIOCS utilizes a linearized model to solve the optimal control objective and constraint functions. Therefore, a mismatch is expected to be present when using the non-linear model as a plant. The standalone BIOCS tracking results for the CO_2 percentage in the outgoing stream and the temperature of the recycled solvent are presented in Figure 9-18 and Figure 9-19, respectively.

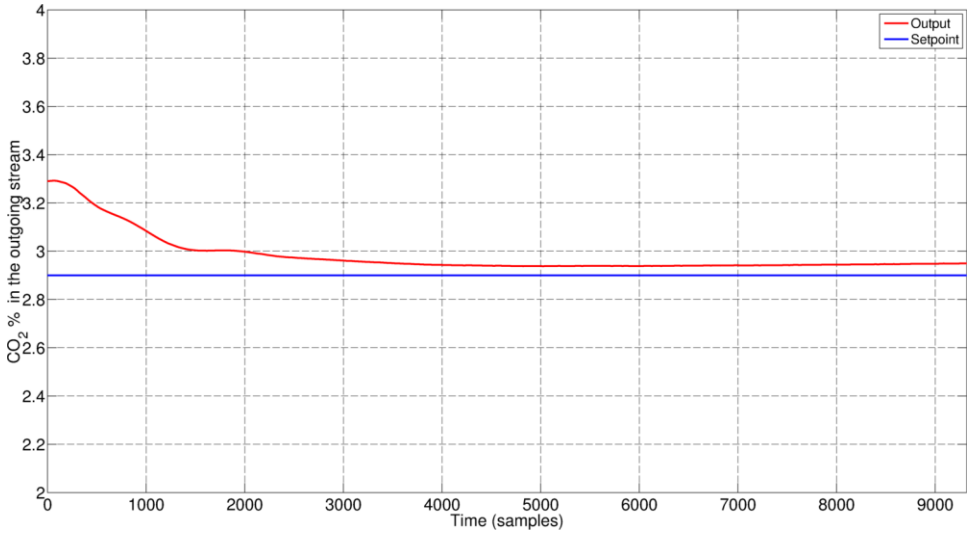

Figure 9-18. CO_2 Percentage in Outgoing Stream Tracking Using BIOCS

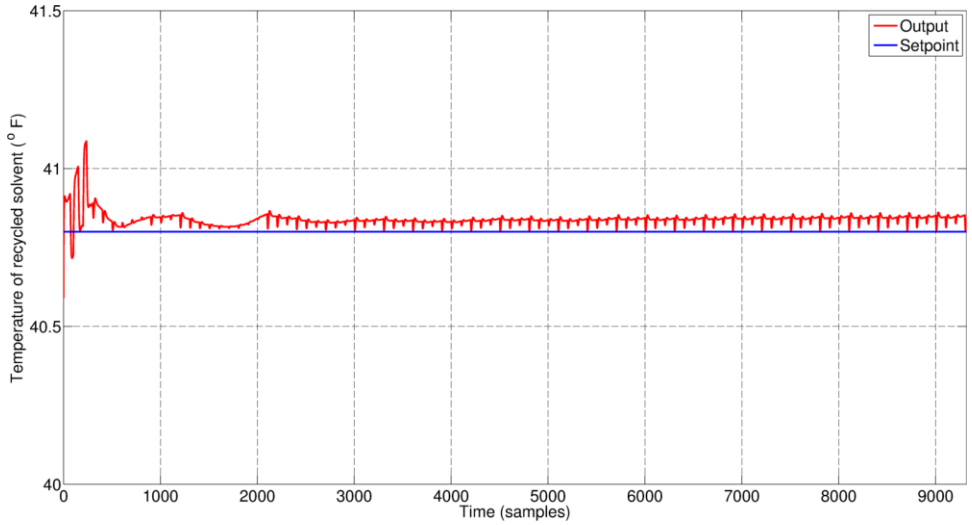

Figure 9-19. Temperature of Recycled Solvent Tracking Using BIOCS

The BIOCS is then augmented with ANN adaptive controller as shown in Figure 9-17. The results of the ANN augmented BIOCS are presented in Figure 9-20 and Figure 9-21. It can be noted that the augmented controller has a better tracking error profile than the standalone version. The CO_2 percentage in the outgoing stream reaches the setpoint with zero tracking error. The temperature of the recycled solvent has a smaller tracking error with smaller variation. This confirms the analysis carried out using the linearized model. More tests are being carried out to further confirm the trend and will be reported in future publications.

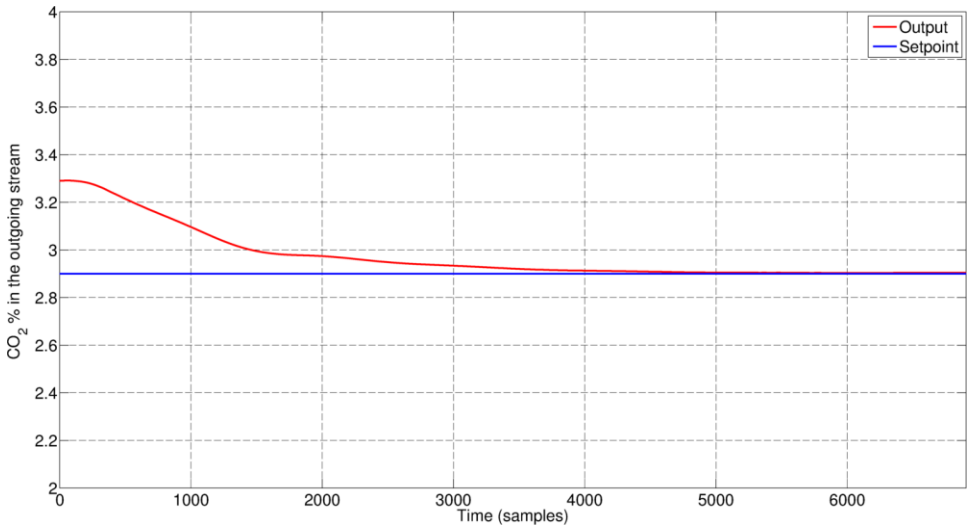

Figure 9-20. CO_2 Percentage in Outgoing Stream Tracking Using BIOCS and ANN

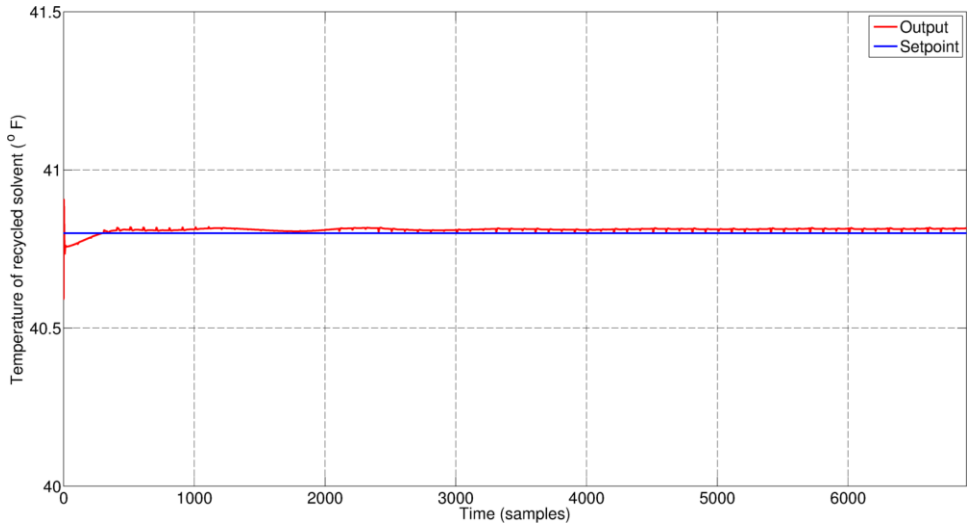

Figure 9-21. Temperature of Recycled Solvent Tracking Using BIOCS and ANN

9.2.3. Demonstration Using the Hyper System

The fuel cell/gas turbine hybrid system considered in this demonstration is a promising technology for the future power generation. The hybrid system is characterized by a high theoretical efficiency as compared to traditional stand-alone turbine or fuel cells configurations. The hybrid system has strongly coupled components, which can lead to control difficulties. Traditional control methods could not provide adequate performance for setpoint tracking and disturbance rejection for such coupled systems [119].

The system presented here is a linear model for the Hyper project [120], [121] managed and run by the National Energy Technology Laboratory (NETL). The project aims at evaluating the dynamic coupling between different fuel cell/gas turbine

configurations. The configuration of the hybrid system used in this study is presented in Figure 9-22.

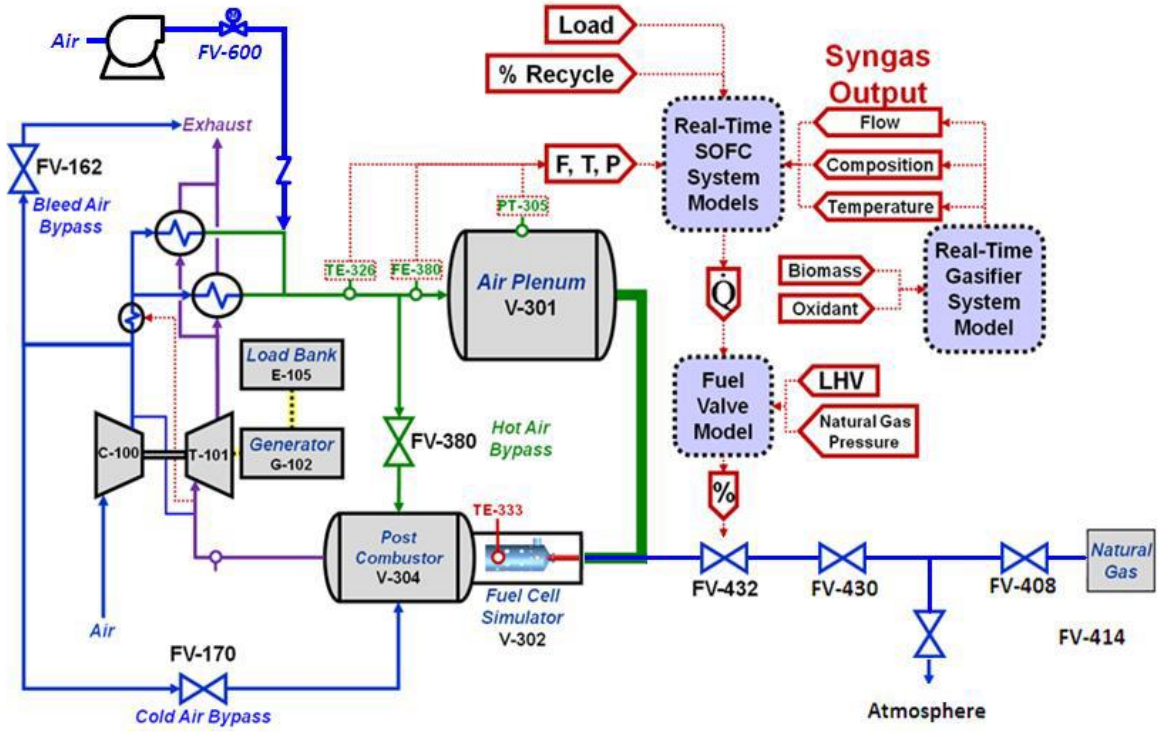

Figure 9-22. Fuel Cells/Gas Turbine Hyper System [119]

The Hyper system uses a combination of software and hardware in order to evaluate the dynamics of a fully integrated system. An online simulation model for the solid oxide fuel cell system, gasifier, and thermal energy storage is coupled to an actual turbine and heat exchangers. The use of the online simulation in the loop facilitates the evaluation of dynamic performance during transient operation, avoiding the destruction of expensive fuel cell equipment. Interested readers are referred to references [121] and [119] for more information about the Hyper project.

A linear multivariable model for the hybrid system was experimentally developed [122]. The model is only representative of the hardware components of the hybrid cycle presented in Figure 9-22. Specifically, only the gas turbine recuperated cycle including the big size volumes of the hybrid configuration was considered during the generation of this model. The modeling process yielded eight transfer functions: turbine speed/electric load (TS/EL), cathode airflow/electric load (CA/EL), turbine speed/cold-air bypass (TS/CAB), cathode airflow/cold-air bypass (CA/CAB), turbine speed/fuel valve (TS/FV),

cathode airflow/fuel valve (CA/FV), turbine speed/hot-air bypass (TS/HAB), and cathode airflow/hot-air bypass (CA/HAB). All these linear transfer functions (as shown in Eq. (9-1)) are first order with delay, with the three defining parameters as listed in Table 9-8.

$$TF(s) = \frac{g}{s - \lambda} e^{\tau s} \quad (9\text{-}1)$$

Table 9-8. Hybrid System Linear Transfer Functions Parameters

Transfer Function	Gain (g)	Pole (λ)	Delay (τ)
TS/EL	-0.25	-0.225	0.1
CA/EL	-0.22	-0.69	0.56
TS/CAB	-0.065	-0.125	0.5
CA/CAB	-1.43	-1.43	0.7
TS/FV	0.17	-0.125	0.1
CA/FV	0.06	-0.16	0.56
TS/HAB	0.03	-0.46	0
CA/HAB	-1.23	-1.43	0.64

A multi-input multi-output state-space controller was previously designed and experimentally tested with promising results [119]. The objective of the controller is to maintain the turbine speed (TS) and the fuel cell cathode airflow (CA) through manipulating the cold air bypass valve and the electric load as presented in Figure 9-23.

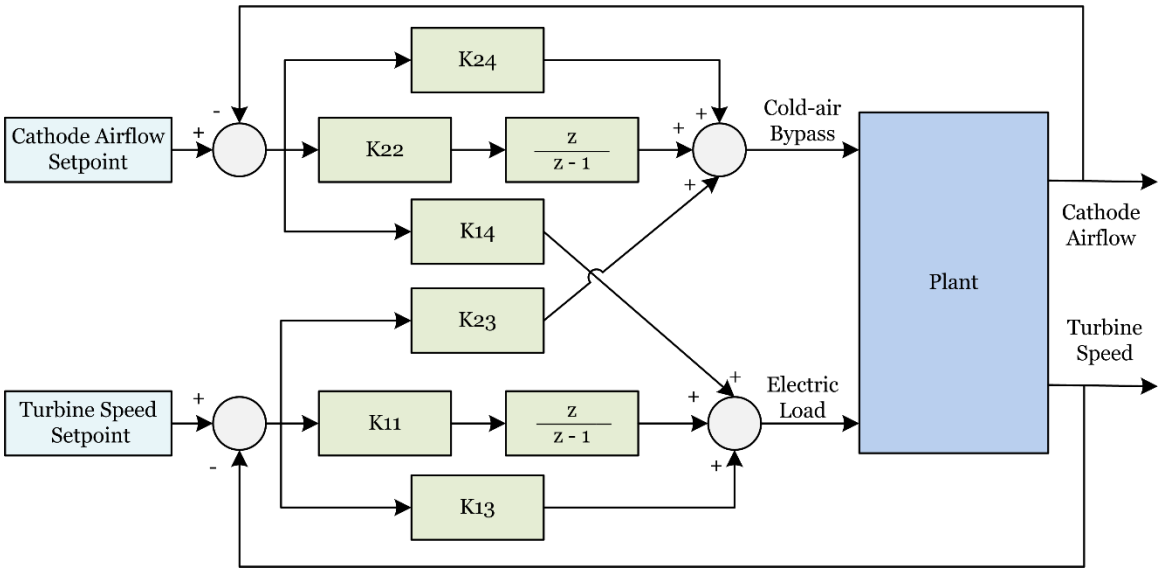

Figure 9-23. Hyper System Baseline Control Laws

In the baseline controller, cross-channel interaction (K23 and K14) between actuators was included to support reducing the hardware coupling effects. The controller was specifically designed to provide desirable performance at nominal conditions around

a fairly linear region of operation. During the experimental implementation, some inconsistency was noticed between simulation results and the hardware performance attributed primarily to modeling uncertainties. It appears that the interaction between system components produces conflicting dynamics for pressure and thermal transients that are difficult to capture using linear system identification techniques. This provided one of the main reasons for investigating the potential of adaptive biomimetic mechanisms for increasing the robustness and the adaptability of baseline control laws.

Both NN-based and AIS-based adaptive control mechanisms presented in Chapter 6 were implemented and tested using the linear hybrid plant models presented Table 9-8. The internal parameters of the adaptive mechanisms were determined heuristically to ensure that baseline performance at nominal conditions is preserved. The testing scenarios include command tracking and disturbance rejection. The commanded inputs are steps of turbine speed and cathode airflow. The disturbances are steps in fuel valve and hot air valve opening.

For all 4 testing scenarios, normal and abnormal conditions were considered. The operation point was considered for a turbine speed (TS) of 40,500 rpm, cathode airflow (CA) of 1 kg/s, with the fuel valve (FV) half-open, and the hot air valve at 40%. The input tracking scenarios consisted of separate step inputs in TS and CA of 500 rpm and 0.2 kg/s, respectively, between t = 50 sec and t = 170 sec. The disturbance rejection tests involved 4% perturbation of FV or HAB valve opening, starting at t = 50 sec for 120 sec. Plant alterations consist of variations of the 3 parameters of the 8 transfer functions. Table 9-9 presents the range of parameters under normal and abnormal conditions.

Table 9-9. Parameter Range for Abnormal Condition Simulation

Transfer Function Parameter	Range
Delay	$[\tau, \quad 45\tau]$
Gain	$[0.0125g, \quad 8g]$
Time Constant	$[0.125\tau_c, \quad 80\tau_c]$

For the evaluation and comparison of alternative control laws, three individual metrics m have been considered: percentage overshoot or maximum tracking error (e_m), mean of the absolute value of tracking errors ($\overline{e}$), and integral of the absolute value of the tracking error (e). For each metric, individual normalized performance indices were defined as:

$$\mathfrak{e}_m = \max[\left(1 - \frac{m}{m_{co}}\right), 0], m = e_m, \bar{e}, or\ e \tag{9-2}$$

where m_{co} is a pre-defined cut-off value. A composite performance vector ($\mathfrak{p}$) was defined as the weighted average of the individual performance indices. If the individual performance vector is defined as:

$$\mathfrak{e} = [\mathfrak{e}_{e_m} \quad \mathfrak{e}_{\bar{e}} \quad \mathfrak{e}_e] \tag{9-3}$$

and the weight vector is expressed as:

$$\mathcal{W} = [w_{e_m} \quad w_{\bar{e}} \quad w_e]^T \tag{9-4}$$

the performance of the controller $\mathfrak{p}$ is defined as:

$$\mathfrak{p} = \mathcal{W}.\mathfrak{e} \tag{9-5}$$

Equal weights were assigned to all three metrics in this study. The performance index was calculated separately for TS and CA errors. A total of 824 cases were simulated equally distributed among TS tracking, CA tracking, FV disturbance rejection, and HAB disturbance rejection. The abnormal condition cases include both individual transfer function parameter alteration as well as multiple alterations.

The results show that the ANN-based adaptive mechanism improves performance at nominal conditions and exhibits superior robustness as compared to the baseline in 85% of cases, on both TS and CA channels. The AIS-based adaptive mechanism exhibits similar performance under nominal conditions and outperforms the baseline in 88% of the cases on the CA channel. Example performance evaluations of the three sets of control laws under nominal conditions are presented in Table 9-10.

Table 9-10. Composite Performance Index (PI) Under Nominal Conditions

Test Case	Baseline		Baseline + ANN		Baseline + AIS	
	TS Channel	CA Channel	TS Channel	CA Channel	TS Channel	CA Channel
TS Tracking	0.84	0.82	0.86	0.85	0.77	0.87
CA Tracking	0.79	0.21	0.80	0.23	0.72	0.48
FV Disturbance	0.24	0.68	0.25	0.74	0.24	0.70
HAB Disturbance	0.91	0.77	0.92	0.80	0.91	0.88

The responses to TS and CA step inputs of the three sets of control laws are presented in Figure 9-24 and Figure 9-25, respectively. The FV and HAB disturbance

rejection capabilities are illustrated in Figure 9-26 and Figure 9-27, respectively. Figure 9-28 presents an example of system response to TS step under increased delay conditions.

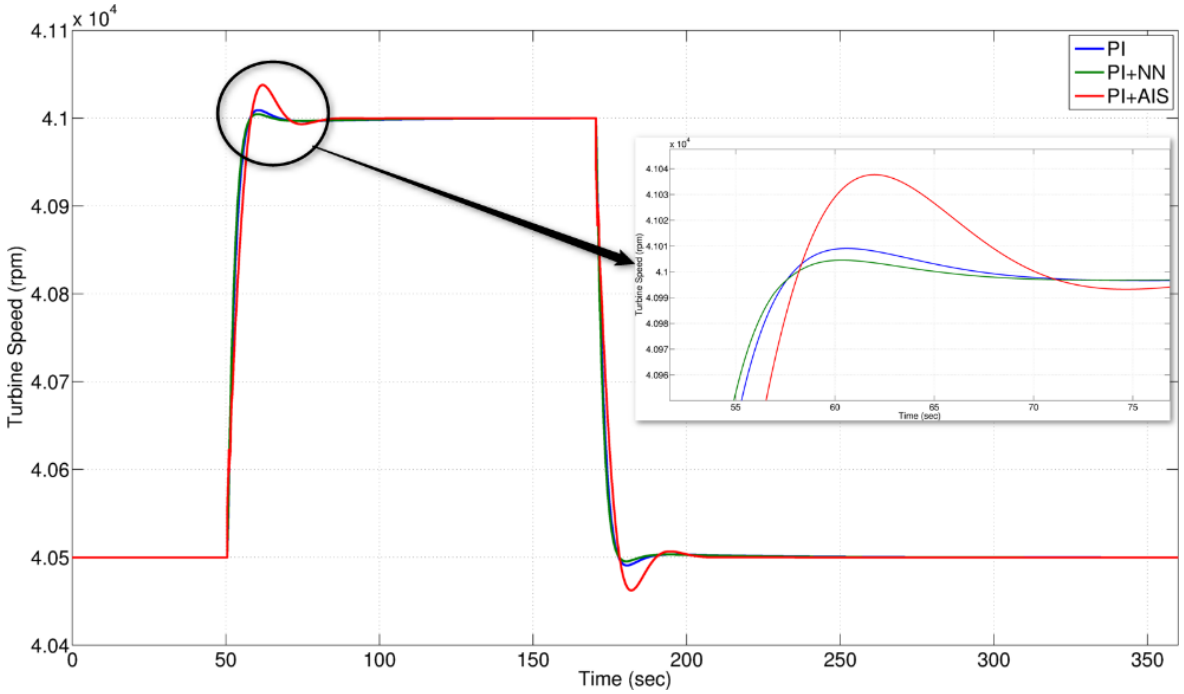

Figure 9-24. Response to TS Step Input Under Nominal Conditions

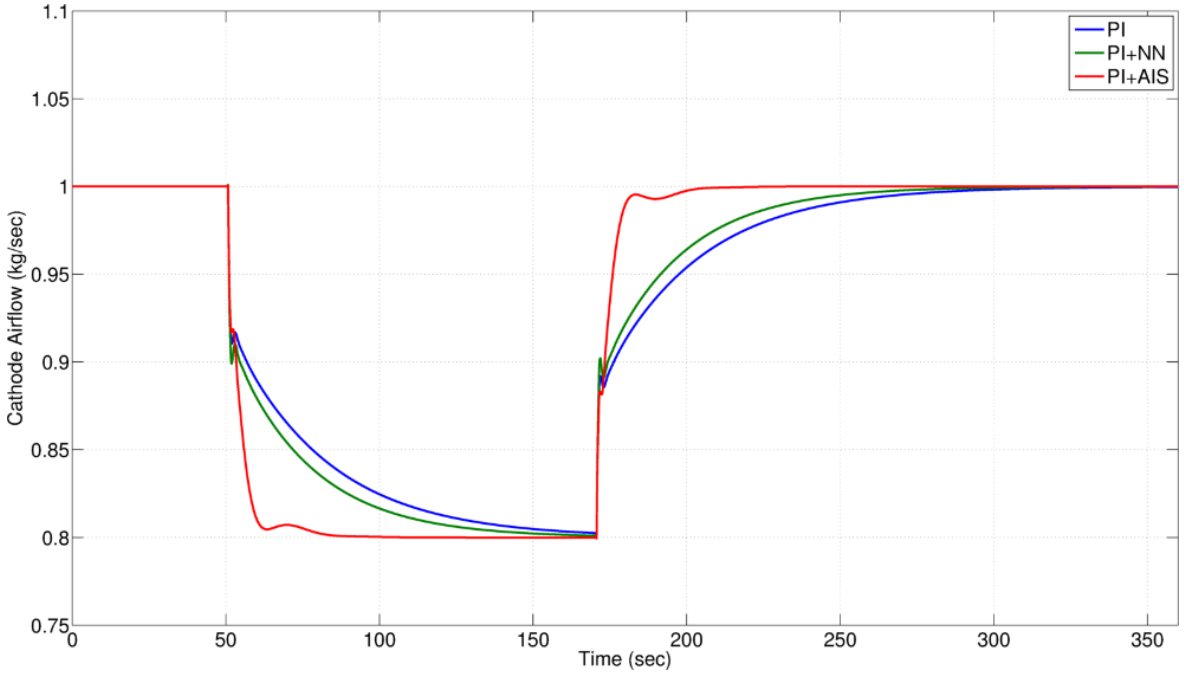

Figure 9-25. Response to CA Step Input Under Nominal Conditions

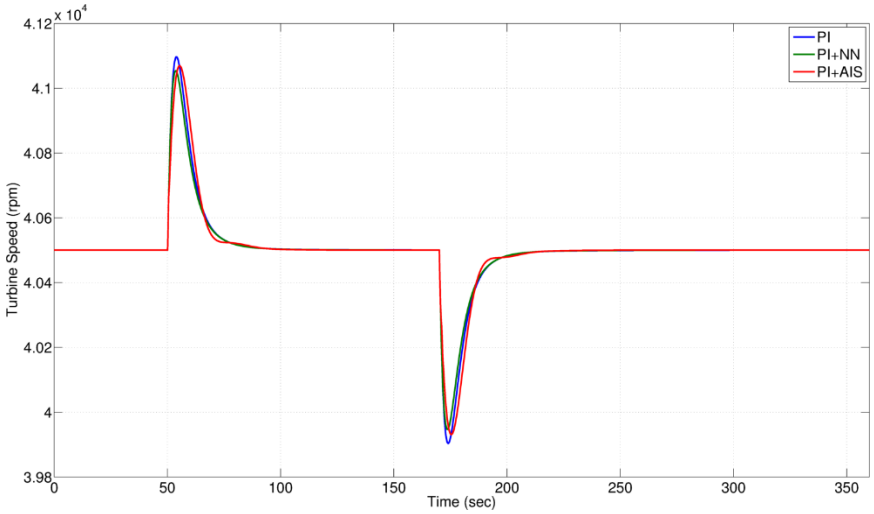

Figure 9-26. TS Response to FV Disturbance Under Nominal Conditions

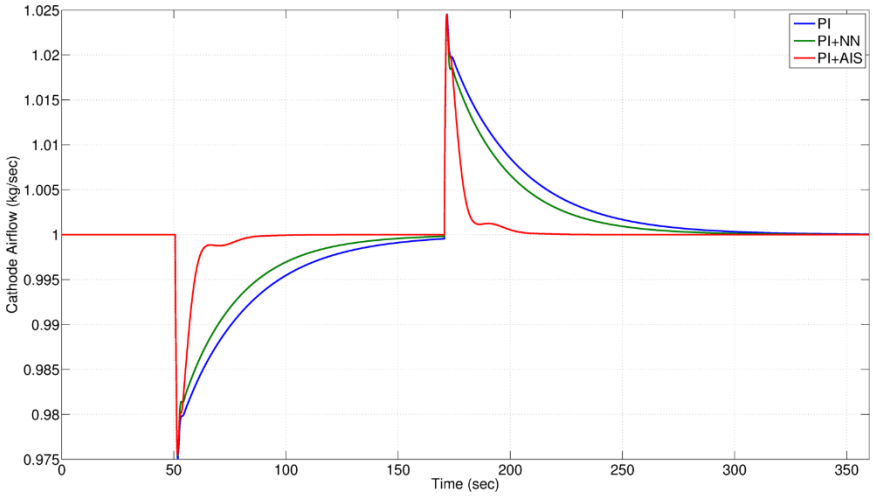

Figure 9-27. CA Response to HAB Disturbance Under Nominal Conditions

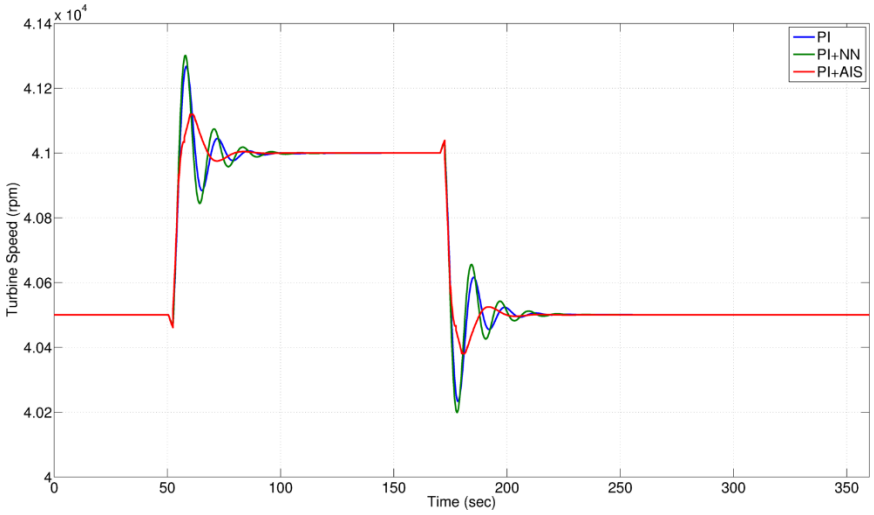

Figure 9-28. Response to TS Step Input Under Abnormal Conditions

9.3. Demonstration of Optimization Scheme

The genetic algorithms described in Chapter 7 were implemented in Matlab® and interfaced with Dynsim® and used to optimize the PID gains of a controller that maintains solvent temperature by manipulating the refrigerant flowrate to the solvent chillers. This is a highly nonlinear problem with large delay due to the large solvent inventory of the system while the rate of change of the refrigeration duty is constrained due to the vapor-compression system. As the solvent becomes colder, more CO_2 is absorbed in the process. As CO_2 is released in the flash vessels, solvent temperature keeps decreasing thus leading to a delayed secondary effect. The process gain of the refrigeration system also changes strongly depending on the operating conditions. Thus oscillatory response is typically observed using non optimal PID control. A five component fitness function was used including rise time (the time required for the process controlled variable to go from 10% to 90% of the desired steady state set point), the maximum of the absolute value of the tracking error calculated after rise time, the mean of the tracking error, standard deviation, and the integral of the absolute value of the tracking error. The overall performance index was computed as the average of these individual performance metrics.

The resolution selected for the three gains was 500, which resulted in a search space of 1.25×10^8 potential solutions. The population consisted of 20 individuals. For the standard algorithm, one point cross-over rate with 30% probability and mutation rate of 60% probability were used. Standard roulette wheel selection approach was adopted for new generation selection. With the stand-alone genetic algorithm, the performance index of the best solution reached 0.94 after 200 generations. The total number of solution explored was 5000.

Within the immunity based enhanced genetic algorithm, the clonal loop was added to the standard algorithm. The loop was performed every generation of 3 times per generation with 6 clones for each individual. The best clone from each individual was then kept to keep the number of individual per generation. Since the clonal loop enhance exploitation, the genetic algorithm cross over rate upped to 70% probability and mutation rate was lowered to 10% probability. Standard roulette wheel selection approach was still used for the new generation selection. The performance index of the best solution reached 0.96 after 75 generations. The total number of solution explored was 3575.

The variation of the overall performance index of the best individual and the average of the population for the standard genetic algorithm is presented in Figure 9-29, and for the clonal loop enhanced algorithm is presented in Figure 9-30. All results represent averages over three different runs. In all runs, the trends were consistent with the averages and the differences between best solutions were not significant. While a complete statistical investigation is still needed, these results from multiple runs provide promise regarding the consistency and relevance of the approach.

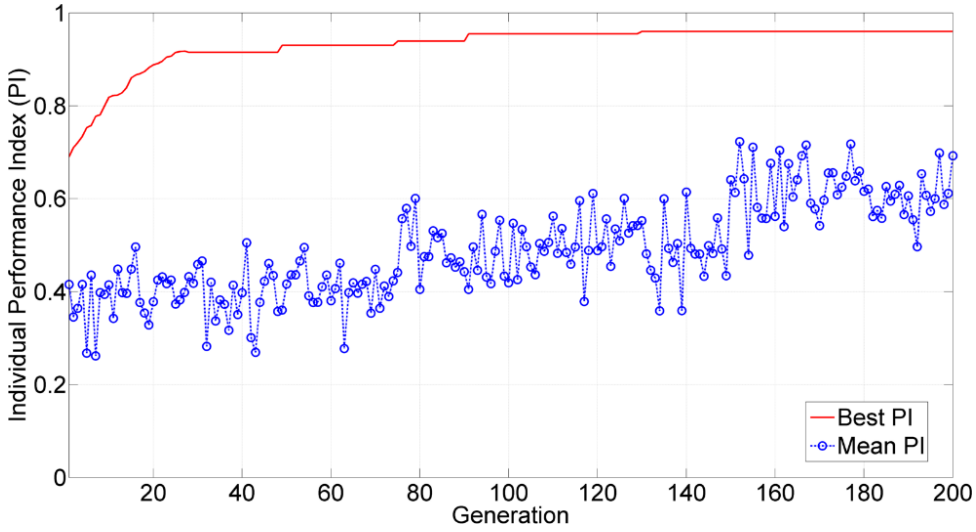

Figure 9-29. Standard Genetic Algorithm - Variation of Performance Index of the Best Individual and the Population Average

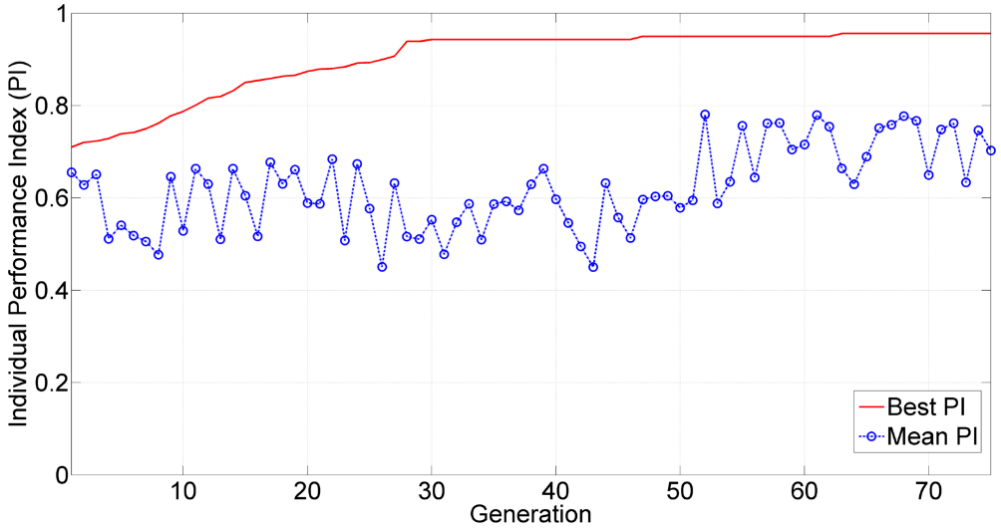

Figure 9-30. Evolutionary Optimization Algorithm with Clonal Loop - Performance Index of the Best Individual and the Population Average

The immunity based enhanced genetic algorithm with clonal loop was tested with added vaccination operator. The operator consists of imposing some gain limits that were expected to induce higher performance index values. As expected, the vaccination operator boosted the performance index of the best individual in the initial population from 0.69 to 0.775. The variation of the overall performance index of the best individual and the average of the population for the enhanced algorithm with vaccination presented in Figure 9-31.

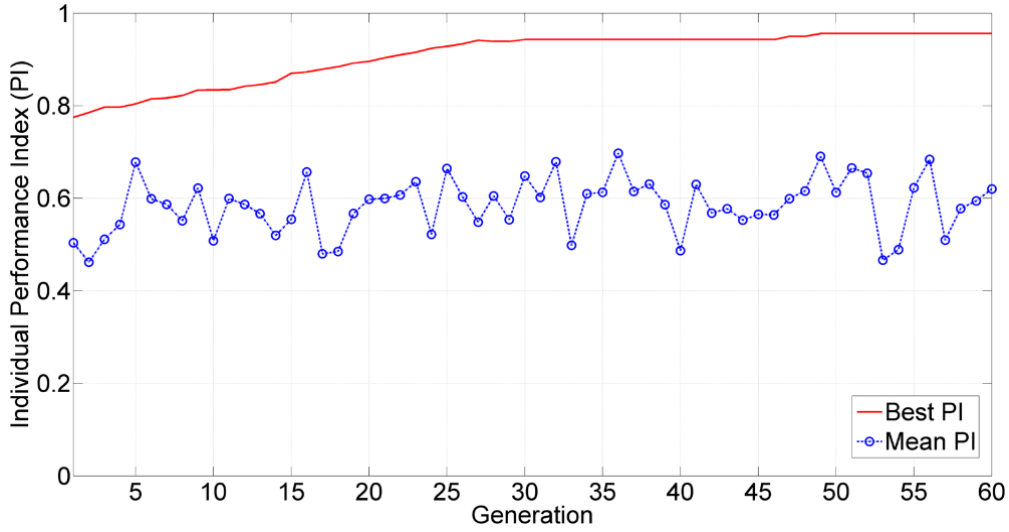

Figure 9-31. Evolutionary Optimization Algorithm with Vaccination - Performance Index of the Best Individual and the Population Average

The immunity based enhanced genetic algorithm with clonal loop was also tested with added seeding operator. The seeding operator consisted of introducing the default PID gains of the targeted controller as well as gains from similar PID controllers from other sections of the plant to form up to 50% of the initial population. The variation of the overall performance index of the best individual and the average of the population for the enhanced algorithm with seeding is presented in Figure 9-32. All results represent averages over three different runs. It is worth mentioning that all other immunity enhanced parameters are similar as before. Figure 9-33 presents the variation of the solvent temperature using the best set of gains. It should be noted that in this example the best individual from all runs was used.

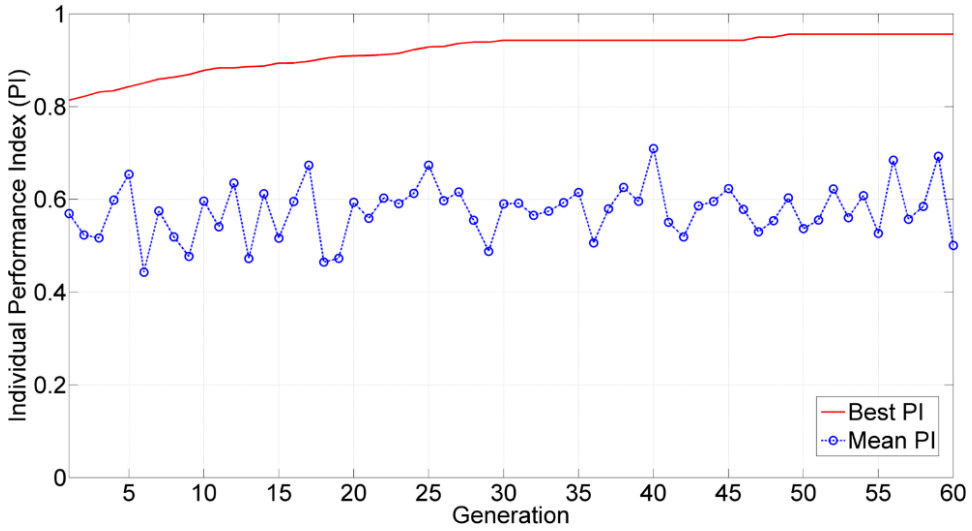

Figure 9-32. Evolutionary Optimization Algorithm with Seeding - Performance Index of the Best Individual and the Population Average

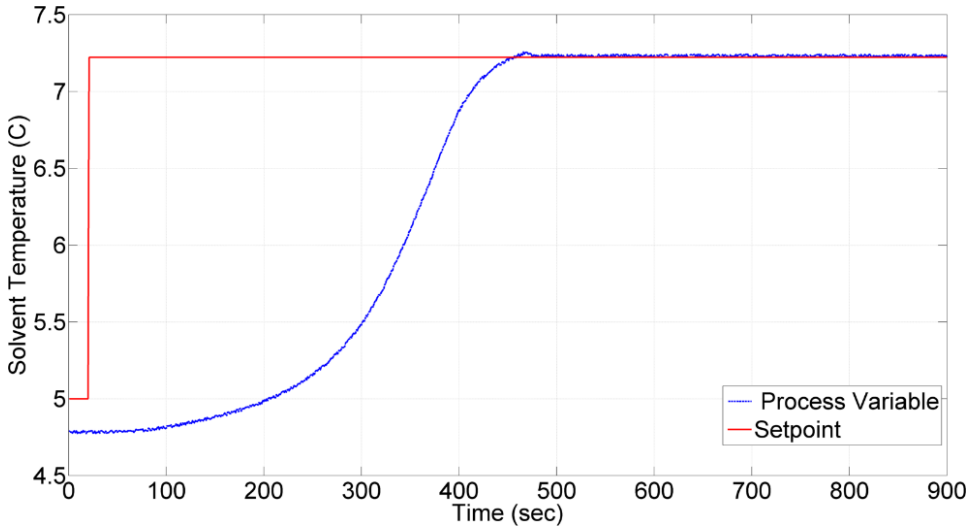

Figure 9-33. Variation of the Solvent Temperature Using the Best Solution

Chapter 10. Developed Tools

This chapter presents a set of tools that have been developed as part of this research effort. These tools were used to implement, test, and modify the proposed algorithms. The tools are designed to provide the future researchers with the means to further test, use, and modify the proposed algorithms with interactive graphical user interfaces and well-documented codes.

10.1. The Self/Non-self Visualization Tool

Visualizing the 2-dimensional sub-selves or projections is very helpful in analyzing how well the selves are generated using the PUA. A simple interactive tool shown in Figure 10-1 was developed to analyze all the generated selves by visualizing the 2-D sub-selves of the AIS for the AGR unit. Test data obtained from simulation within Dynsim® can be projected on these sub-selves to analyze and validate them.

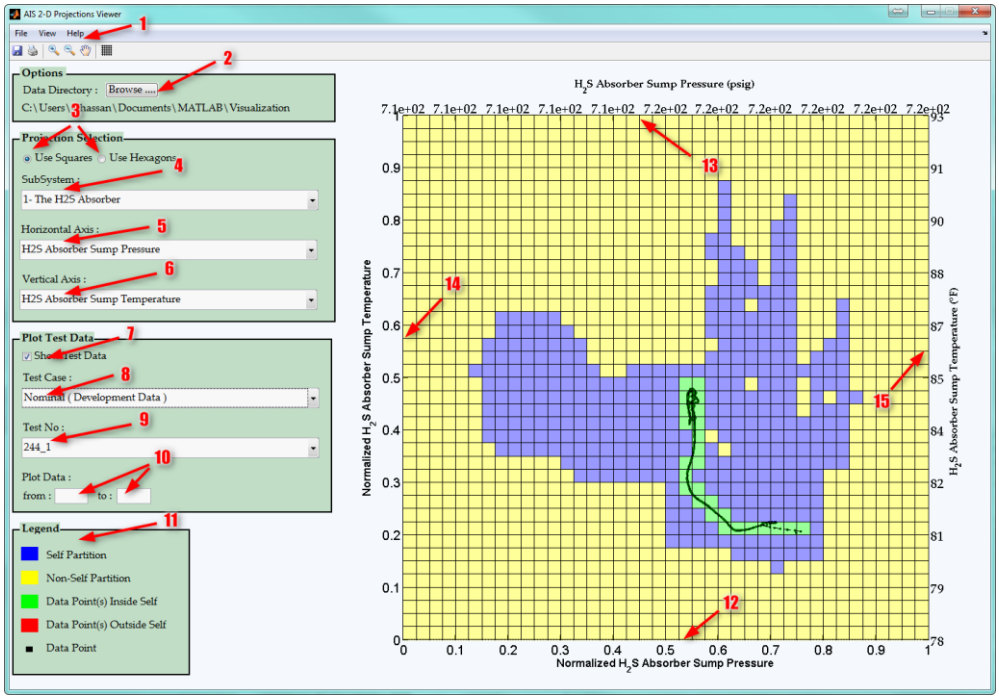

Figure 10-1. AIS Selves and Non – Selves 2-D Projections Viewer

The tool has the following main components as illustrated in Figure 10-1:

1. The main toolbar: contains the options to save, print, zoom, pan, and view the grid

2. The data directory: contains a push button to navigate to the data location and a text box to display the current directory.

3. The projection type radio buttons: are used to switch between squares projections type and hexagons projection type.

4. The subsystem menu: contains the list of 22 subsystems of the AGR unit.

5. The horizontal axis menu: each subsystem contains a certain number of selected features associated with that subsystem. Use this menu to select the feature that will be plotted on the horizontal axis.

6. The horizontal axis menu: each subsystem contains a certain number of selected features associated with that subsystem. Use this menu to select the feature that will be plotted on the horizontal axis.

7. Display test data checkbox: once this checkbox is selected, the selected data set will be displayed on the plot.

8. The test case data menu: the data is organized into four sets:

- Nominal (Validation): This set of data was collected under nominal condition and used to validate the created projections.
- Nominal (Development Data): This set of data was collected under nominal conditions and used to build the self-clusters.
- Failure (System Malfunction): This set of data was collected in the presence of several system malfunctions.
- Abnormal Condition (Constrained Violation): This set of data was collected for the purpose of self building; however, they violate one or more nominal conditions constraints.

9. The test number menu: in each data group, there are many tests. Each test is numbered or named differently, the content of this menu is dynamically changed based on the test case selection.

10. The data limit: (not operational)

11. The plot legend.

12. The primary horizontal axis: this axis displays the horizontal selected feature. The default limits are zero and one. Use the pan feature to drag the figure and see data beyond those limits.

13. The secondary horizontal axis: this axis displays the actual scale for the selected horizontal feature along with its units.

14. The primary vertical axis: this axis displays the vertical selected feature. The default limits are zero and one. Use the pan feature to drag the figure and see data beyond those limits.

15. The secondary horizontal axis: this axis displays the actual scale for the selected horizontal feature along with its units.

The tool uses a Microsoft excel file to populate the GUI menus, axis labels, feature minimum and maximum values, and feature units, which make the tool easy to modified and used for future research without modifying the tool Matlab® code.

10.2. The ACDIE Interface

WVU ACDIE is an interactive simulation environment that was developed to integrate the ACDIE algorithms coded in Matlab® and the AGR unit model in Dynsim®. The purpose of this environment is to provide a user-friendly interface to test and demonstrate the effectiveness of the ACDIE scheme presented in Chapter 5. This improves the generality and flexibility of the environment by easily allowing for tuning of the DC algorithm parameters and allowing the selection of various AC types and severities.

The GUI is automatically launched once the AGR simulation model is loaded in Dynsim®. Once launched, Matlab® opens the interface for the configuration of DC mechanism parameters, which is presented in Figure 10-2. It contains the parameters settings for the algorithm. This graphical user interface (GUI) is designed to accept only valid parameter settings.

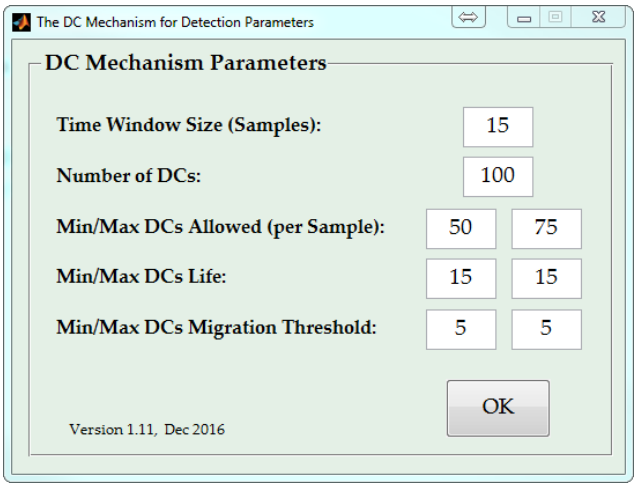

Figure 10-2. GUI for Parameter Setting of the DC Mechanism

After pressing OK, two message boxes will appear (Figure 10-3 and Figure 10-4) to remind the user to use the "Load Full" (LF) command and to load a set of "Initial Conditions" (IC) after selecting a simulation scenario. Those steps are necessary to apply the selected abnormal conditions and to start the simulation from an initial conditions known to belong to the self.

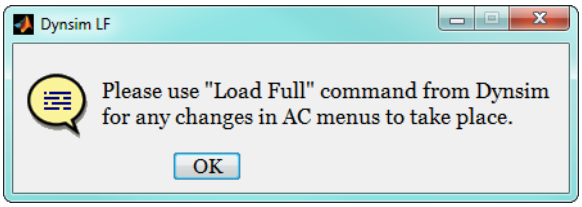

Figure 10-3. Load Full Command Request Message Box

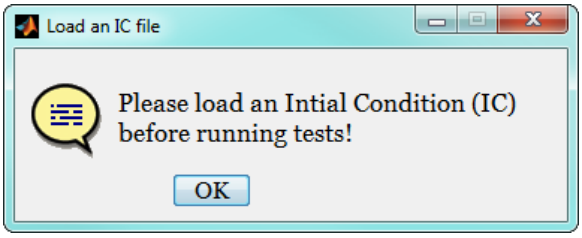

Figure 10-4. Load Initial Conditions Request Message Box

After pressing "OK" to both messages, an interactive GUI will appear as showed in Figure 10-5. The GUI allows the user to select the AC type and severity using two drop-down menus. All the AC and severities listed in Table 8-3 were implemented and tested using this interface.

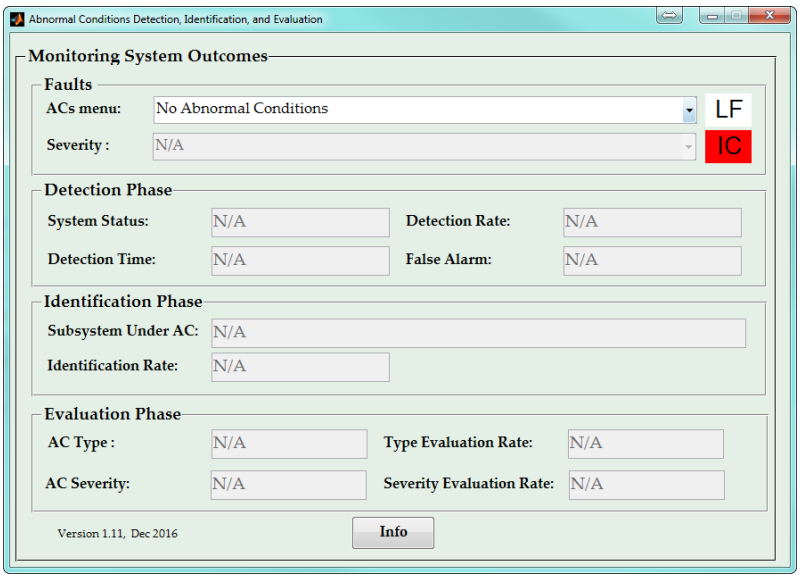

Figure 10-5. ACDIE Interactive GUI

The following steps are necessary to run a scenario using the ACDIE:

1. Use the "ACs menu" to select an AC to simulate
2. Use the "Severity" menu to select the severity of the simulated AC
3. Use load full function from Dynsim® software
4. Load an IC (Note: the IC data should be included in the self)
5. Run the simulation

Once the simulation is ran, the GUI displays the artificial DC algorithm outcomes for all detection, identification, and evaluation phases as follows:

1. The "System Status" indicates the algorithm detection outcome for the mode of operation. It displays "Normal Operation" or "Abnormal Operation".
2. The "Detection Time" text box displays DT
3. The "Detection Rate" text box displays DR
4. The "False Alarm Rate" text box displays FA
5. The "Subsystem Under AC" displays the name of the subsystem primarily affected by the detected AC.
6. The "Identification Rate" text box displays IR
7. The "AC Type" displays the type of the detected AC
8. The "Type Evaluation Rate" displays TER
9. The "AC Severity" displays the estimated AC severity level
10. The "Severity Evaluation Rate" displays SER

10.3. The Evolutionary Optimization Environment Interface

WVU evolutionary optimization environment is an interactive simulation environment that was developed to integrate the optimization algorithms built in Matlab® with the AGR unit model in Dynsim®. The purpose of this environment is to provide a user-friendly interface between the simulation environment and the optimization algorithms. This improves the generality and flexibility of the environment as well as the GA by easily allowing for tuning of gains of controllers with various structures in response to different scenarios. The general optimization scenario and

specific algorithms and parameters are setup by the user through a series of Matlab® GUI menus.

Once the optimization-enabled AGR Dynsim® model is loaded, it launches Matlab® and a series of Matlab® GUI. The GUI allow the user to select the optimization problem and set the selected optimization parameters as follows:

The first GUI (Figure 10-6) contains two menus. The first menu contains the options to select the optimization problem type, control system parameters vs. control system setpoints optimization. The second menu contains the options to select the algorithm type, standard GA vs. immunity enhanced algorithm.

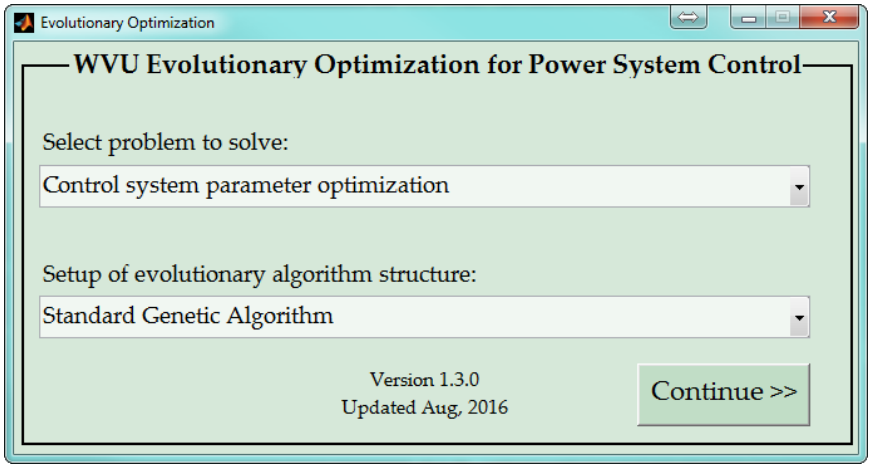

Figure 10-6. GUI for Control System Optimization Problem and Algorithm Selection

The second GUI (Figure 10-7) is for setting the initial population and convergence parameters. The "Initial population options" drop-down menu allows for new population generation or loading an existence population data file. The codes verify the loaded data and only allow compatible data to be loaded. The rest of GUI is self-explanatory.

If the immunity enhanced algorithm option is selected, the next GUI will serve for the clonal loop parameters setting (Figure 10-8). The GUI allows the selection of the clonal loop mutation algorithm, the number of the clones per individual, and the hyper-mutation rate. The GUI also allows for the setting of the iteration based convergence algorithm.

The next GUI (Figure 10-9) is for the selection algorithm, mutation rate, and cross over rate setting. The GUI also contains the setting to alter the simulation time of each individual for fitness function evolution.

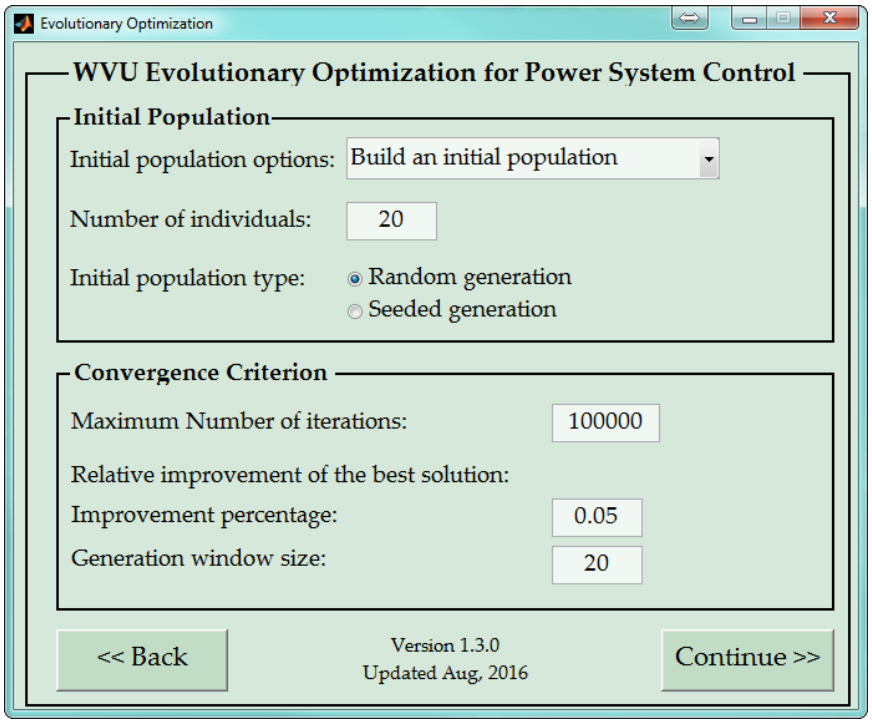

Figure 10-7. GUI for GA Initial Population and Convergence Parameter Setting

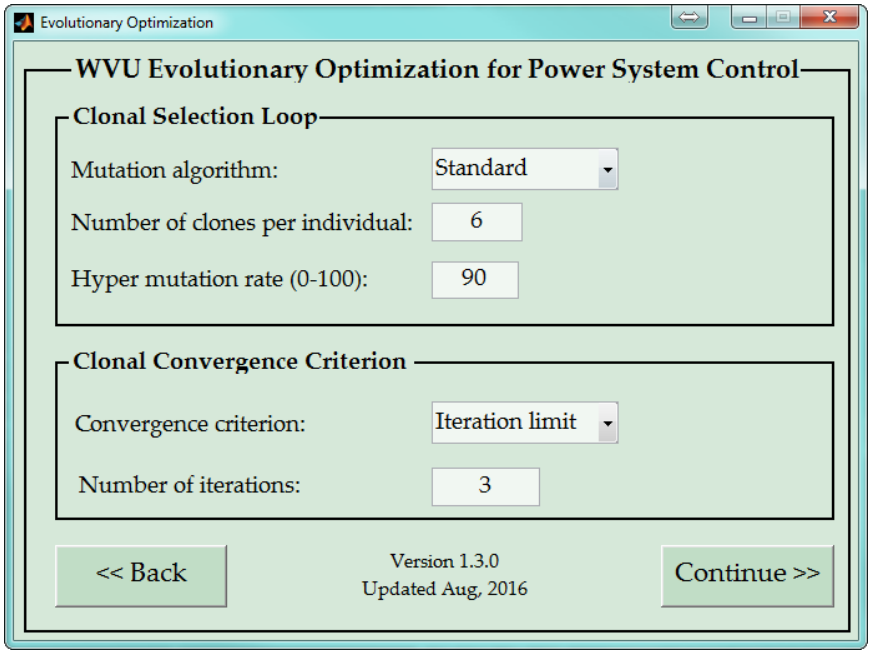

Figure 10-8. GUI for Setup of Parameter of Clonal Selection Loop

The final GUI (Figure 10-10) is for display purposes. It allows the user to select what Matlab® displays during running the algorithm as well as the plot displayed after one or more convergence criteria have been met.

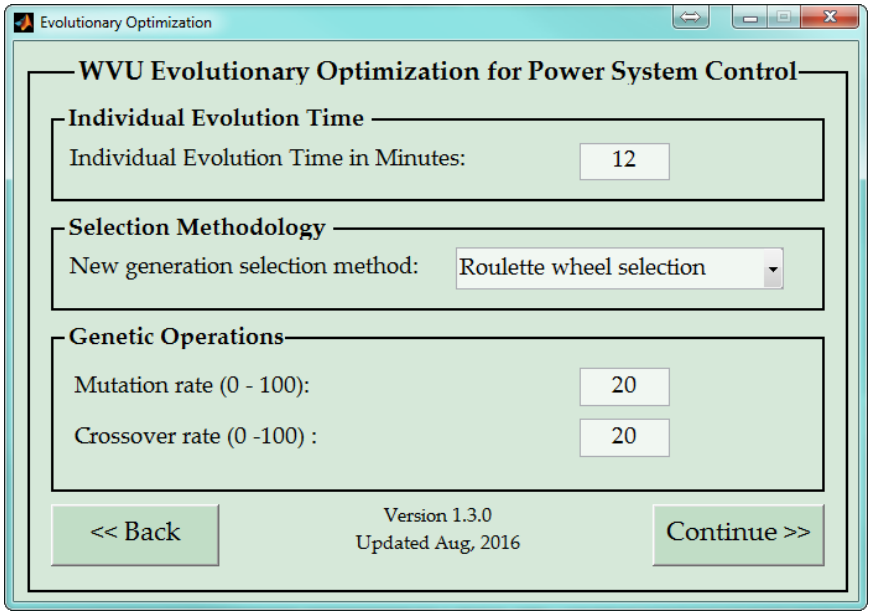

Figure 10-9. GUI for Setup of GA Selection Method and Alteration Parameters

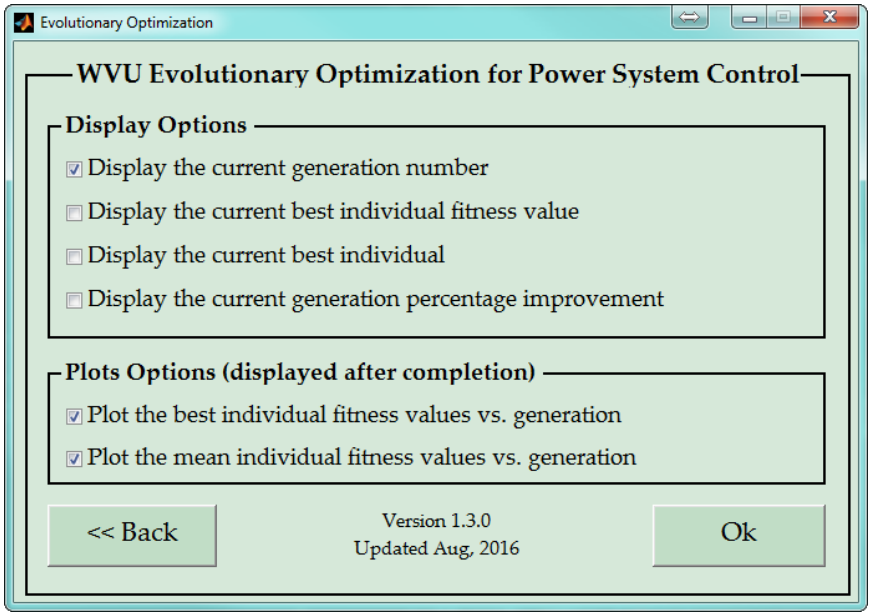

Figure 10-10. GUI for Output Display

Chapter 11. Conclusion and Future Work

The dissertation presented a comprehensive monitoring and control methodology for advanced power plant using artificial intelligence techniques. The methodology is comprehensive in the sense that it offers the framework for addressing all system components, all AC (including known and unknown ones), and all major components of the system monitoring and control process (AC detection, identification, evaluation, and accommodation).

The proposed methodology is highly integrated because one single package relying on few shared concepts and mechanisms can solve all components of the system monitoring and control process. Conceptually, the methodology is data-driven, which makes it easy to develop, implement, and operate. Unlike alternative approaches, it does not require a sophisticated mathematical apparatus prone to difficult development and computational and numerical issues. The methodology provides robustness and adaptability because it allows for relatively simple updating, extension, and re-structuring of the self/non-self throughout the life cycle of the system.

Within this research, the PUA was formulated as an alternative approach for self/non-self generation within the AIS paradigm. Generating the self with the proposed approach is less computationally expensive than clustering for the same number of features and with similar resolution. The generation of the non-self is implicit and the computationally intensive effort through clustering algorithms is no longer necessary. The PUA allows for higher – dimensional self projection use, which results in a minimal overlapping between self and non-self and therefore better detection results. The use of higher dimensional self projects also facilitated the introduction of training-free AC identification using the subsystem pattern approach.

A modified artificial DC algorithm has been formulated and successfully tested within the AIS paradigm to address the specific characteristics of power plant health monitoring using the HMS strategy. The modified algorithm is capable of addressing specific issues related to the application of AIS to modern power plants, including non-uniform dimensionality of the self/non-self projections, extended hidden non-self regions within lower order projections, and unbalanced distribution of projections that capture the occurrence of ACs.

Two different adaptive mechanisms have been presented in this study, namely the ANN-based and AIS-based mechanisms. These novel biomimetic adaptive approaches have been implemented to augment baseline power plant control laws for increased robustness under ACs. Testing with different baseline control laws and different plant models has demonstrated the promising capabilities of the proposed adaptive mechanisms.

Several immunity-inspired mechanisms have been proposed to enhance the performance of standard genetic algorithms. The immunity based mechanisms show the capability to accelerate the search and fine-tune high performance solutions. An interactive framework that interfaces Matlab® and Dynsim® to provide a powerful computational environment for power plant control optimization using evolutionary techniques has also been developed.

All the proposed algorithms were successfully illustrated using the AGR unit high fidelity model built in Dynsim®. The implementation results of the proposed DC mechanism for ACDIE show the high capability of the approach in in detecting, identifying, and evaluating all AC considered while minimizing the false alarms rate to 0% for all tested nominal conditions.

In considering future research for the monitoring and control framework improvements and extension, some recommendations are proposed as following:

- Using the PUA for self representation, only hyper-rectangles and hyper-hexagons were considered. The use of other shapes are still possible and it should be considered.
- The current framework provides excellent ACDIE outcomes; however, it would be interesting to investigate the possibility of building and using an AIS capable of predicting AC occurrences.
- The current DC algorithm is only used for ACDIE, but the algorithm has the potential to address the AC accommodation as well.
- The analysis presented in this research was carried out using simulation tests. The results obtained are encouraging to continue this research effort by implementing the proposed methodology on an actual system.

Chapter 12. References

[1] B. Rukes and R. Taud. "Status and perspectives of fossil power generation". *Energy* vol. *29*, no. *12–15*, pp. *1853-1874*, 2004.

[2] W. L. Luyben, B. D. Tyreus and M. L. Luyben, *Plantwide Process Control*. McGraw-Hill, 1999.

[3] G. P. Rangaiah and V. Kariwala, *Plantwide Control: Recent Developments and Applications*. John Wiley & Sons, 2012.

[4] D. Bhattacharyya *et al.*, "Development of Integrated Biomimetic Framework with Intelligent Monitoring, Cognition, and Decision Capabilities for Control of Advanced Energy Plants", *Presented at 2017 NETL Project Review Meeting for Crosscutting Research, Gasification Systems, and Rare Earth Elements Research Portfolios, Pittsburgh, PA, March 20-23,* 2017.

[5] V. Venkatasubramanian *et al.* "A review of process fault detection and diagnosis: Part I: Quantitative model-based methods." *Computers & chemical engineering*, vol. *27*, no. *3*, pp. *293-311*, 2003.

[6] V. Venkatasubramanian *et al.* "A review of process fault detection and diagnosis: Part II: Qualitative models and search strategies." *Comput. Chem. Eng.* vol. *27*, no. *3*, pp. *313-326*. 2003.

[7] V. Venkatasubramanian *et al.* "A review of process fault detection and diagnosis: Part III: Process history based methods." *Computers & chemical engineering*, vol. *27*, no. *3*, pp. *327-346*, 2003.

[8] M. Bouzenita *et al.* "A Neural Modeling Approach for the Diagnosis of Elementary Failure Modes in Industrial Plants."

[9] R. M. Behbahani, H. Jazayeri-Rad and S. Hajmirzaee. "Fault detection and diagnosis in a sour gas absorption column using neural networks." *Chem. Eng. Technol.*, vol. *32*, no. *5*, pp. *840-845*, 2009.

[10] M. Namdari, H. Jazayeri-Rad and S. Hashemi. "Process fault diagnosis using support vector machines with a genetic algorithm based parameter tuning." *Journal of Automation and Control*, vol. *2*, no. *1*, pp. *1-7*, 2014.

[11] I. Yélamos *et al.* "Performance assessment of a novel fault diagnosis system based on support vector machines." *Computers & chemical engineering*, vol. *33*, no. *1*, pp. *244-255*, 2009.

[12] Q. Jiang, X. Yan and W. Zhao. "Fault detection and diagnosis in chemical processes using sensitive principal component analysis." *Ind. Eng. Chem. Res.*, vol. *52*, no. *4*, pp. *1635-1644*, 2013.

[13] L. Rusinov, N. Vorobiev and V. Kurkina. "Fault diagnosis in chemical processes and equipment with feedbacks." *Chemometrics Intellig. Lab. Syst.*, vol. *126*, pp. *123-128*, 2013.

[14] X. Chen and X. Yan. "Using improved self-organizing map for fault diagnosis in chemical industry process." *Chem. Eng. Res. Design*, vol. *90*, no. *12*, pp. *2262-2277*, 2012.

[15] Russell, Evan L., Leo H. Chiang, and Richard D. Braatz. *Data-driven methods for fault detection and diagnosis in chemical processes*. Springer Science & Business Media, 2012.

[16] P. Paul *et al.* "Sensor network design for maximizing process efficiency: An algorithm and its application." *AIChE Journal*, vol. *61*, no. *2*, pp. *464-476*, 2015.

[17] M. Bhushan, S. Narasimhan and R. Rengaswamy. "Robust sensor network design for fault diagnosis." *Comput. Chem. Eng.*, vol. *32*, no. *4*, pp. *1067-1084*, 2008.

[18] Dasgupta, Dipankar, editor, *Artificial Immune Systems and Their Applications*, Springer, 1999.

[19] R. Turton *et al.* (2016, March 3). AVESTAR [Webpage]. Available: http://avestar.wvu.edu/.

[20] T. J. Kindt *et al. Kuby immunology*. Macmillan, 2007.

[21] C. Janeway *et al. Immunobiology: The Immune System in Health and Disease*. New York: Garland Science, 2005.

[22] Dasgupta, Dipankar, and Fernando Nino. *Immunological computation: theory and applications*. CRC press, 2008.

[23] D. L. Hamilos. "Antigen presenting cells." *Immunologic Research*, vol. *8*, no. *2*, pp. *98-117*, 1989.

[24] R. M. Steinman. (2016, May 9). *Introduction to Dendritic Cells Laboratory of Cellular Physiology and Immunology* [Webpage]. Available: http://lab.rockefeller.edu/steinman/dendritic_intro/.

[25] C. S. William *et al.* "Positive and negative selection of an antigen receptor on T cells in transgenic mice." *Nature,* vol. *336,* no. *6194,* pp. *73-76*. 1988.

[26] N. Toma, S. Endo and K. Yamanda. "Immune algorithm with immune network and MHC for adaptive problem solving." *Conf. Proc. IEEE International Conf. SMC*, vol. *4*, pp. *271-276*, 1999.

[27] N. Mitsumoto *et al.* "Control of the distributed autonomous robotic system based on the biologically inspired immunological architecture" *Conf. Proc. IEEE International Conf. Robotics and Automation,* vol. *4,* 1997.

[28] N. Mitsumoto *et al.* "Self-organizing multiple robotic system (a population control through biologically inspired immune network architecture)." *Conf. Proc. IEEE International Conf. Robotics and Automation,* 1997.

[29] Y. Watanabe et al. "Emergent construction of a behavior arbitration mechanism based on the immune system." *Adv. Rob.*, vol. *12*, no. *3*, pp. *227-242*, 1997.

[30] S. Forrest *et al.* "A change-detection algorithm inspired by the immune system." *IEEE Transactions on Software Engineering*, 1995.

[31] D. Dasgupta and S. Forrest. "Novelty detection in time series data using ideas from immunology." *Proc. of the international conference on intelligent systems*, pp. *82-87*, 1996.

[32] L. Zhi-tang, L. Yao and W. Li. "A novel fuzzy anomaly detection algorithm based on artificial immune system." Presented at *High-Performance Computing in Asia-Pacific Region,* 2005.

[33] G. Luh, C. Wu and W. Cheng. "Artificial immune regulation (AIR) for model-based fault diagnosis." Presented at *The 3rd International Conference on Artificial Immune Systems*. 2004

[34] C. Laurentys, R. M. Palhares and W. M. Caminhas. "A novel artificial immune system for fault behavior detection." *Expert Syst. Appl.* vol. 38, no. 6, pp. *6957-6966*, 2011.

[35] A. B. Serapião, J. R. Mendes and K. Miura. "Artificial immune systems for classification of petroleum well drilling operations." Presented at *the 6th International Conference on Artificial Immune Systems (ICARIS)*, 2007.

[36] J. Timmis and T. Knight. "Artificial immune systems: Using the immune system as inspiration for data mining." *Data Mining: A Heuristic Approach*, pp. *209-230*, 2001.

[37] J. E. Hunt and A. Fellows. "Introducing an immune response into a CBR system for data mining." *BCS ESG*, 1996.

[38] P. Hajela and J. S. Yoo. "Immune network modelling in design optimization. Presented at *New Ideas in Optimization*, 1999.

[39] C. A. C. Coello *et al.* "Use of emulations of the immune system to handle constraints in evolutionary algorithms." 2001.

[40] S. Endoh, N. Toma and K. Yamada. "Immune algorithm for n-TSP." *Conf. Proc.* IEEE International Conference On *Systems, Man, and Cybernetics*, 1998.

[41] J. D. Farmer, N. H. Packard and A. S. Perelson. "The immune system, adaptation, and machine learning." *Physica D.*, vol. *22,* no. 1, pp. *187-204*, 1986.

[42] A. Watkins, J. Timmis and L. Boggess. "Artificial immune recognition system (AIRS): An immune-inspired supervised learning algorithm." *Genetic Programming and Evolvable Machines*, vol. *5*, no. *3*, pp. *291-317*, 2004.

[43] S. Forrest, S. A. Hofmeyr and A. Somayaji. "Computer immunology." *Commun. ACM.,* vol. *40*, no. *10*, pp. *88-96*, 1997.

[44] J. O. Kephart. "A biologically inspired immune system for computers." Presented at *Artificial Life IV: Proc. of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*. 1994.

[45] G. B. Lamont, R. E. Marmelstein and D. A. Van Veldhuizen. "A distributed architecture for a self-adaptive computer virus immune system." Presented at *New Ideas in Optimization*. 1999.

[46] W. Sun *et al.* "An artificial immune network with diversity for pattern recognition." Presented at *SICE Annual Conference*, 2004.

[47] H. Dai *et al.* "Clonal selection theory based artificial immune system and its application." Presented at *Neural Information Processing*. 2006.

[48] T. Knight and J. Timmis. "AINE: An immunological approach to data mining." *Proc. 2001 IEEE international conference on data mining*, pp. *297-304*, 2001.

[49] Yanfei Zhong et al. "An unsupervised artificial immune classifier for multi/hyperspectral remote sensing imagery." *IEEE Transactions on Geoscience and Remote Sensing*, vol. *44*, no. *2*, pp. *420-431*, 2006.

[50] M. Su *et al.* "A neuro-fuzzy approach for compensating color backlight images." *Neural Process Letters*, vol. *23*, no. *3*, pp. *273-287*, 2006.

[51] S. Forrest *et al.* "Self-nonself discrimination in a computer". *Research in Security and Privacy*, 1994.

[52] A. Somayaji *et al.* "Principles of a computer immune system." *Proc. of the 1997 workshop on New security paradigms*, pp. *75-82*, 1998.

[53] J. Kim and P. J. Bentley. "An evaluation of negative selection in an artificial immune system for network intrusion detection." *Proc. of the 3rd Annual Conference on Genetic and Evolutionary Computation*, pp. *1330-1337,* 2001.

[54] J. Kim and P. J. Bentley. "Towards an artificial immune system for network intrusion detection: An investigation of clonal selection with a negative selection operator." *Proc. of the 2001 Congress on Evolutionary Computation*, 2001.

[55] J. Kim and P. J. Bentley. "Towards an artificial immune system for network intrusion detection: An investigation of dynamic clonal selection." *Proc. of the 2002 Congress Presented at Evolutionary Computation*, 2002.

[56] T. S. Guzella, T. A. Mota-Santos and W. M. Caminhas. "A novel immune inspired approach to fault detection." *Proc. of Artificial Immune Systems,* 2007.

[57] J. Greensmith, U. Aickelin and S. Cayzer. "Introducing dendritic cells as a novel immune-inspired algorithm for anomaly detection." *Proc. of Artificial Immune Systems*, 2005.

[58] P. Matzinger. "The danger model: A renewed sense of self." *Science*, vol. *296*, no. *5566*, pp. *301-305*, 2002.

[59] J. Greensmith and U. Aickelin. "The deterministic dendritic cell algorithm." Proc. of Artificial Immune Systems, 2008.

[60] M. Mokhtar *et al.* "A modified dendritic cell algorithm for on-line error detection in robotic systems." *Proc. of IEEE Congress on Evolutionary Computation*, 2009.

[61] J. C. L. Pinto and F. J. Von Zuben. "Fault detection algorithm for telephone systems based on the danger theory." *Proc. of Artificial Immune Systems*, 2005.

[62] Y. Shu and J. Zhao. "Fault diagnosis of chemical processes using artificial immune system with vaccine transplant." *Ind. Eng. Chem. Res.*, vol. *55*, no. *12*, pp. *3360-3371*, 2015.

[63] J. Zhao *et al.* "An online fault diagnosis strategy for full operating cycles of chemical processes." *Ind. Eng. Chem. Res.*, vol. *53*, no. *13*, pp. *5015-5027*, 2013.

[64] M. G. Perhinschi, H. Moncayo and J. Davis. "Integrated framework for artificial immunity-based aircraft failure detection, identification, and evaluation." *J. Aircraft*, vol. *47*, no. *6*, pp. *1847-1859*, 2010.

[65] M. Perhinschi *et al.* "Development of an immunity-based framework for power plant monitoring and control." *Advanced Chemical Engineering Research*, vol. *4*, no. *1*, 2015.

[66] H. Moncayo, M. Perhinschi and J. Wilburn. "Aircraft failure detection and identification over an extended flight envelope using an artificial immune system." *The Aeronautical Journal, Royal Aeronautical Society*, vol. 115, no. 1163, 2011.

[67] H. Moncayo, M. G. Perhinschi and J. Davis. "Aircraft failure detection and identification using an immunological hierarchical multiself strategy." *Journal of Guidance, Control, and Dynamics*, vol. *33,* no. *4*, pp. *1105-1114*, 2010.

[68] D. Al Azzawi, M. G. Perhinschi and H. Moncayo. "Dendritic cell mechanism for aircraft immunity-based failure detection and identification." Presented at and *Proc. of the AIAA Guidance, Navigation, and Control Conference, Boston, MA.,* 2013.

[69] D. A. Azzawi, M. G. Perhinschi and H. Moncayo. "Artificial dendritic cell mechanism for aircraft immunity-based failure detection and identification." *Journal of Aerospace Information Systems*, vol. 11, no. 7, pp. *467-481,* 2014.

[70] K. Takashi and T. Yamada. "Application of an immune feedback mechanism to control system." *JSME International Journal Series C Mechanical Systems, Machine Elements and Manufacturing*, vol. *41*, no. 2, pp. *184-191*, 1998.

[71] J. Lee *et al.* "Clonal selection algorithms for 6-DOF PID control of autonomous underwater vehicles." *Proc. of Artificial Immune Systems*, 2007.

[72] K. Kalmanje and J. Neidhoefer. "Immunized adaptive critic for an autonomous aircraft control application." Presented at *Artificial Immune Systems and their Applications*, 1999.

[73] K. K. Kumar and J. Neidhoefer. "Immunized adaptive critics for level 2 intelligent control." Presented at *Systems, Man, and Cybernetics*, 1997.

[74] K. Krishnakumar and J. Neidhoefer. "Immunised neurocontrol." *Expert Syst. Appl.*, vol. *13*, no. *3*, pp. *201-214*, 1997.

[75] H. Moncayo *et al.* "UAV adaptive control laws using non-linear dynamic inversion augmented with an immunity-based mechanism." *Proc. of AIAA Guidance, Navigation, and Control Conference*, p. *4678*, 2012.

[76] A. E. Perez *et al.* "A bio-inspired adaptive control compensation system for an aircraft outside bounds of nominal design." *Journal of Dynamic Systems, Measurement, and Control*, vol. *137*, no. *9*, 2015.

[77] C. L. Karr, K. Nishita and K. S. Graham. "Adaptive aircraft flight control simulation based on an artificial immune system." *Applied Intelligence,* vol. *23*, no. *3*, pp. *295-308*, 2005.

[78] A. Togayev *et al.* "Immunity-based accommodation of aircraft subsystem failures." *Aircraft Eng. Aerospace Technol.*, vol. *89*, no. *1*, pp. *164-175*, 2017.

[79] J. Brownlee. "Autonomous Distributed Control in the Immune System using Diffuse Feedback." *CIS Technical Report*, 2007.

[80] L. N. De Castro and F. J. Von Zuben. "Learning and optimization using the clonal selection principle." *IEEE Transactions on Evolutionary Computation*, vol. *6*, no. *3*, pp. *239-251*, 2002.

[81] L. N. de Castro and J. Timmis. "An artificial immune network for multimodal function optimization." *Proc. of the 2002 Congress on Evolutionary Computation,* 2002.

[82] J. Timmis, C. Edmonds and J. Kelsey. "Assessing the performance of two immune inspired algorithms and a hybrid genetic algorithm for function optimisation." *Proc. of the 2004 Congress on Evolutionary Computation*, 2004.

[83] F. Freschi and M. Repetto. "Multiobjective optimization by a modified artificial immune system algorithm." *Proc. of International Conference on Artificial Immune Systems*. 2005.

[84] X. Yao, Y. Liu and G. Lin. "Evolutionary programming made faster." *IEEE Transactions on Evolutionary Computation*, vol. *3*, no. *2*, pp. *82-102*, 1999.

[85] E. Clark, A. Hone and J. Timmis. "A markov chain model of the b-cell algorithm." *Proc. of International Conference on Artificial Immune Systems*. 2005.

[86] K. Wojdan, K. Swirski and T. Chomiak. "Immune inspired system for chemical process optimization using the example of a combustion process in a power boiler." *Proc. of International Conference on Intelligent Systems Applications to Power Systems*, 2007.

[87] F. Esponda, S. Forrest and P. Helman. "A formal framework for positive and negative detection schemes." *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions*, vol. *34*, no. *1*, pp. *357-373*, 2004.

[88] M. Perhinschi *et al*. "Generation of artificial immune system antibodies using raw data and cluster set union." *International Journal of Immune Computation*, vol. *2*, no. *1*, pp. *1-15*, 2014.

[89] J. MacQueen. "Some methods for classification and analysis of multivariate observations." *Proc. of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

[90] J. Davis, M. G. Perhinschi and H. Moncayo. "Evolutionary algorithm for artificial-immune-system-based failure-detector generation and optimization." *Journal of Guidance, Control, and Dynamics*, vol. *33*, no. *2*, pp. *305-320*, 2010.

[91] D. Al Azzawi. "Aircraft Abnormal Conditions Detection, Identification, and Evaluation using Innate and Adaptive Immune Systems Interaction." *Ph.D. Dissertation, West Virginia University, United State*, 2014.

[92] G. Al-Sinbol and M. G. Perhinschi. "Generation of power plant artificial immune system using the partition of the universe approach." *International Review of Automatic Control (IREACO)* vol. *9*, no. *1*, pp. *40-47*, 2016.

[93] G. Al-Sinbol, M. Perhinschi, D. Bhattacharyya, "Power Plant Abnormal Condition Detection Using the Artificial Immune System Paradigm" *Poster Presented at 2016 AIChE Annual Meeting, San Francisco, CA.,* November 13-18, 2016

[94] D. A. Azzawi, M. G. Perhinschi and H. Moncayo. "Artificial dendritic cell mechanism for aircraft immunity-based failure detection and identification." *Journal of Aerospace Information Systems,* vol. *11*, no. *7*, pp. *467-481*, 2014.

[95] M. G. Perhinschi and G. Al-Sinbol "Artificial dendritic cell algorithm for advanced power system monitoring" *International Review of Automatic Control (IREACO),* vol. *9*, no. *5*, pp. *330-340*, 2016.

[96] C. M. Bishop. *Pattern Recognition and Machine Learning*, Springer, 2006

[97] G. Al-Sinbol, M. Perhinschi, "Development of an Artificial Immune System for Power Plant Abnormal Condition Detection, Identification, and Evaluation", *submitted to International Review of Automatic Control (I.RE.A.CO.),* Feb 2017.

[98] M. G. Perhinschi, H. Moncayo and D. Al Azzawi. "Development of immunity-based framework for aircraft abnormal conditions detection, identification, evaluation, and accommodation." *Proc. of the AIAA Guidance, Navigation, and Control Conference, Boston, MA.*, 2013.

[99] H. Y. Moncayo. "Immunity-Based Detection, Identification, and Evaluation of Aircraft Sub-System Failures," *Ph.D. Dissertation, West Virginia University, United States,* 2009.

[100] M. G. Perhinschi *et al.* "A simulation environment for testing and research of neurally augmented fault tolerant control laws based on non-linear dynamic inversion." *Proc. of the AIAA Modeling and Simulation Technologies Conference,* 2004.

[101] A. Calise, S. Lee and M. Sharma. "Direct adaptive reconfigurable control of a tailless fighter aircraft." *Proc. of Guidance, Navigation, and Control Conference and Exhibit*, 1998.

[102] D. Bhattacharyya D *et al.* "Development of Integrated Biomimetic Framework with Intelligent Monitoring, Cognition, and Decision Capabilities for Control of Advanced Energy Plants", *Presented at 2017 NETL Project Review Meeting for Crosscutting Research, Gasification Systems, and Rare Earth Elements Research Portfolios, Pittsburgh, PA.,* March 20-23, 2017

[103] K. Takahashi and T. Yamada. "Application of an immune feedback mechanism to control systems." *JSME International Journal Series C Mechanical Systems, Machine Elements and Manufacturing,* vol. *41*, no. *2*, pp. *184-191*, 1998.

[104] W. Chen, J. Zhou and H. Wei. "Compensatory controller based on artificial immune system." *Proc. of the 2006 IEEE International Conference on Mechatronics and Automation*, 2006.

[105] H. Mo and L. Xu. "A kind of improved immune controller." *Proc. of International Conference Mechatronics and Automation (ICMA)*, 2009.

[106] X. Gao *et al.* "Simulation and research of fuzzy immune adaptive PID control in coke oven temperature control system." *Proc. of Sixth World Congress on Intelligent Control and Automation (WCICA)*, 2006.

[107] L. Davis. *Handbook of Genetic Algorithms*, 1991.

[108] D. E. Goldberg and J. H. Holland. "Genetic algorithms and machine learning." *Mach. Learning*, vol. *3*, no. *2*, pp. *95-99*, 1988.

[109] Z. Michalewicz and S. J. Hartley. "Genetic algorithms data structures= evolution programs." *Mathematical Intelligencer*, vol. *18*, no. *3*, pp. *7*, 1996.

[110] G. Al-Sinbol *et al.*, "Evolutionary Optimization of Power Plant Control System Using Immunity-inspired Algorithms" *International Review of Chemical Engineering (I.RE.CH.E.)*, vol. *9*, no.1, 2017.

[111] D. Bhattacharyya, R. Turton and S. E. Zitney. "Steady-state simulation and optimization of an integrated gasification combined cycle power plant with CO2 capture." *Ind. Eng. Chem. Res.*, vol. *50*, no. *3*, pp. *1674-1690*, 2010.

[112] P. Mobed et al. "Optimal sensor placement for fault diagnosis using magnitude ratio." *Ind. Eng. Chem. Res.*, vol. *54*, no. *38*, pp. *9369-9381*, 2015.

[113] S. E. Zitney *et al.* "AVESTAR center for operational excellence of clean energy plants and invensys DYNSIM OTS / EYESIM ITS integration." Presented at *SimSci-Esscor North American User Group & Technical Support Symposium*, 2012.

[114] S. E. Zitney *et al.* "AVESTAR center: Dynamic simulation-based collaboration toward achieving opertional excellence for IGCC plants with crbon capture." *Proc. of the 29th Annual International Pittsburgh Coal Conference*, 2012.

[115] Schneider Electric (12/01/2016). SimSci DYNSIM [Webpage]. Available: http://software.schneider-electric.com/products/simsci/design/dynsim/.

[116] [109] D. Jones *et al.* "Plant-wide control system design: Primary controlled variable selection." *Comput. Chem. Eng.*, vol. *71*, pp. *220-234*, 2014.

[117] G. Mirlekar *et al.* "Design and implementation of a Biologically-Inspired Optimal Control Strategy (BIO-CS) for chemical process control". *Accepted for publication in Industrial Engineering and Chemistry Research (I&ECR) journal.*

[118] G. Mirlekar *et al.* "A Biologically-Inspired Optimal Control Strategy (BIO-CS) for hybrid energy systems". *To appear in Proc. of the 2017 American Control Conference (ACC).*

[119] P. Pezzini *et al.* "Multi-coordination of actuators in advanced power systems." *Proc. of ASME Turbo Expo 2015: Turbine Technical Conference and Exposition,* 2015.

[120] D. Hughes *et al.* "Transient behavior of a fuel cell/gas turbine hybrid using hardware-based simulation with a 1-D distributed fuel cell model." Presented at *10th International Colloquium on Environmentally Preferred Advanced Power Generation (ICEPAG 2010), Costa Mesa, CA.,* Feb. 2010.

[121] D. Tucker *et al.* Technical description of the hybrid performance project test facility and system model.

[122] P. Pezzini *et al.* "Decentralized control strategy for fuel cell turbine hybrid systems." Presented at I*SA Power Industry Division Symposium, ISA-PWID2014-Journal of Engineering for Gas Turbine and Power*, 2014.